

RD-A139 604

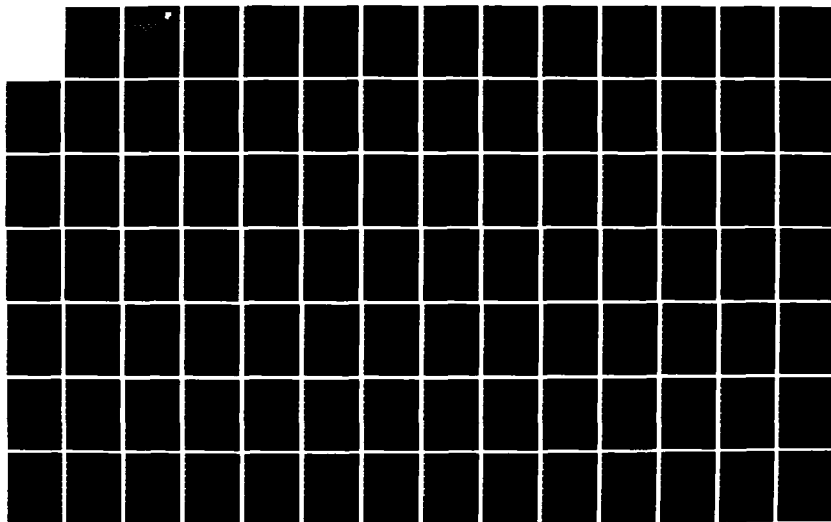
MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND
ELECTRON TRANSPORT VOLUME 2(U) ARCON CORP WALTHAM MA
S WOOLF ET AL JAN 84 RADC-TR-83-313-VOL-2
F19628-78-C-0188

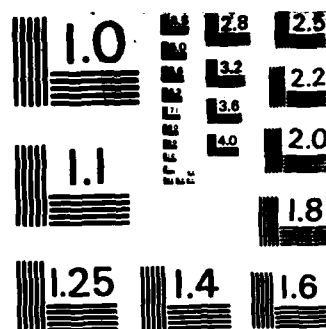
1/5

UNCLASSIFIED

F/G 9/2

NL





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

12
RADC-TR-83-313, Vol II (of two)
Final Technical Report
January 1984

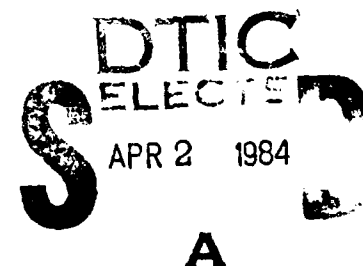


MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND ELECTRON TRANSPORT

Arcon Corporation

**Peter P. Wintersteiner
Stanley Woolf**

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED



**ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441**

DTIC FILE COPY

84 03 30 006

AD A139604

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-83-313, Vol II (of two) has been reviewed and is approved for publication.

APPROVED:



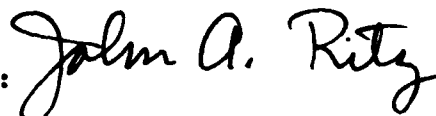
PETER D. GIANINO
Project Engineer

APPROVED:



HAROLD ROTH, Director
Solid State Sciences Division

FOR THE COMMANDER:



JOHN A. RITZ
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (ESO), Hanscom AFB MA 01731. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-83-313, Vol II (of two)	2. GOVT ACCESSION NO. AD-P139604	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND ELECTRON TRANSPORT		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report 1 Feb 78 - 30 Sep 82
7. AUTHOR(s) Stanley Woolf Peter P. Wintersteiner		6. PERFORMING ORG. REPORT NUMBER N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Arcon Corporation 260 Bear Hill Road Waltham MA 02154		8. CONTRACT OR GRANT NUMBER(s) F19628-78-C-0188
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ESO) Griffiss AFB NY 13441		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2306J230
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same		12. REPORT DATE January 1984
		13. NUMBER OF PAGES
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same		
18. SUPPLEMENTARY NOTES RADC Project Engineer: Peter D. Gianino (ESO)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Dielectric Waveguides Multilayered Waveguides Electron Transport Coupled Waveguides Integral Equation Technique Energy Deposition Propagation Variational Principle Charge Deposition Modal Properties Photorefractive Effect Dose-Depth Distribution Crosstalk Phase Conjugate Optics (Cont'd on reverse)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) In this report we will describe the techniques and computer programs developed to analyze various problems. The scope of our work extended into three technical areas: 1) the study of dielectric waveguide modes and arrays of dielectric waveguides. 2) the study of a mechanism for producing diffraction gratings in crystals exhibiting photorefractivity. 3) the problem of electron scattering and transport in solids.		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 19 Cont'd (Key Words)

Method of Discrete Ordinates

Multigroup Transport

Monte Carlo Method

Multiple Scattering

Polymer Irradiation by Electrons

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

- TRUNCN -

Accession For	
NTIS GRA&I	
DTIC TAB	
Unannounced	
Justification	
By	
Distribution	
Availability	
Avail	
Dist	Spec
A-1	

PROGRAM TRUNCN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)

PREPARED BY PETER P. WINTERSTEINER, ARCON CORP: COMPLETION DATE 11/30/78
REVISED FOR IMPROVED EFFICIENCY REVISION COMPLETE 6/ 5/79
FURTHER REVISIONS REVISION COMPLETE 1/ 7/80
FURTHER REVISIONS REVISION COMPLETE 5/28/81
FURTHER REVISIONS REVISION COMPLETE 11/ 5/81

PROGRAM TRUNCN FINDS THE MODES OF A DIELECTRIC WAVEGUIDE IN THE
WEAKLY-GUIDING APPROXIMATION BY SATISFYING THE CONDITIONS $\text{DET}(R) = 0$
OR $\text{DET}(I) = 0$. R AND I ARE MATRICES OBTAINED BY TRUNCATING (AT
NOT TERMS) THE INFINITE SET OF INFINITE LINEAR EQUATIONS FOR THE FIELD
EXPANSION COEFFICIENTS (C OR D). R AND I ARE ACTUALLY LINE
INTEGRALS WHICH ARE EVALUATED BY CHEBYSHEV QUADRATURE. UP TO 20 SEPARATE
CASES CAN BE HANDLED IN A SINGLE JOB, SO LONG AS NOT IS KEPT FIXED.
NOT MAY BE BETWEEN 1 AND 7.

INTEGER S,SC
REAL N,NL
DIMENSION PR(10),TABLE(51,2),INDEX(20)
DIMENSION STORE(200,16),INDX(7),GO(7,7),C(7)
DIMENSION ROOT(20,13),ROOT(20,3),CFF(20,8,7)
COMMON/A,B,P2,PI,NOT
COMMON/B/STORE,INDX,GO,C
DATA (PR(1),1-1.31/10H CIRCLE,10HN ELLIPSE,10H SQUARE, /
DATA (PR(1),1-4.7/10H S' ELLIPSE,10H RECTANGLE,6HCOSINE,6H SINE /
DATA (PR(1),1-8.9/10H PLIPED,10H CUSPED /

LIST OF VARIABLES WITH SINGLE ASSIGNED PURPOSES

- ACC ACCURACY TO WITHIN WHICH THE ROOT LOCATION PROCEDURE FINDS P2
- B NORMALIZED FREQUENCY (READ IN)
- BP VALUE OF B IS SZ.NE.1 (SEE COMMENTS)
- C VECTOR OF COEFFICIENTS IN THE FIELD EXPANSION (DESIRED RESULT)
- CFF ARRAY FOR STORING VALUES OF C FOR DIFFERENT CASES
- DET VALUE OF THE DETERMINANT OF THE MATRIX GO
- DRDP DIRHO/D(PHI) ON THE PERIMETER OF THE GUIDE
- GD MATRIX OF COEFFICIENTS (RLS OR TLS)
- ID # OF FUNCTION EVALUATIONS DURING ROOT-LOCATION PROCEDURE
- IDIA DIAGNOSTIC CODE (SEE COMMENTS) (READ IN)
- INDEX ARRAY FOR STORING TABLE INDICES NEAR WHICH ROOTS OCCUR
- INDX ARRAY FOR STORING VALUES OF MATRIX INDICES USED
- IROOT ARRAY FOR STORING INFORMATION USED IN PRINTING RESULTS AT THE END
- JDEXT COUNTER TO KEEP TRACK OF THE NUMBER OF SEPARATE CASES COMPLETED
- L INDEX OF FIRST TERM IN FIELD EXPANSION (READ IN)
- LLL VALUE OF L FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
- LLS VALUE OF S FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
- N SUPERELLI-SE INDEX (REAL) (READ IN)
- NL VALUE OF N FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
- NOP # OF INTEGRATION POINTS (READ IN)
- NDT # OF TERMS IN EXPANSIONS, SIZE OF DETERMINANT (READ IN)
- NP2 # OF VALUES OF P2 TAKEN TO CONSTRUCT TABLE (READ IN)
- NR # OF ROOTS FOUND
- PHI ANGLE IN RHO, PHI) COORDINATE SYSTEM
- PI
- P1 LARGEST VALUE OF P2 USED IN TABLE (READ IN)
- PMAX SMALLEST VALUE OF P2 USED IN TABLE (READ IN)
- PMIN
- PR ARRAY FOR STORING ALPHANUMERIC PROGRAM INFORMATION
- P2 PROPAGATION CONSTANT (DESIRED RESULT)
- R ASPECT RATIO (READ IN)

[illegible]

```

NOT = NUMBER OF TERMS TAKEN BEFORE TRUNCATION
AT TACY OF HALF-INTERVAL SEARCH IS 10*(1-S)
NP1 = # OF POINTS IN THE INITIAL TABLE
NIP = # OF INTEGRATION POINTS IN THE RANGE (0,PI/2)
IDAG = DIAGNOSIS CODE 0 = NORMAL CONDITION (DEFAULT)
1 = TABLE, MATRIX, COEFF SUMMARY PRINTED
2 = SUMMARY PRINTED WITH MORE SIG FIGURES
3 = DETAILS OF ROOT-LOCATION PRINTED

* IF THE CROSS-SECTION IS CIRCULAR, THE PHI-INTEGRATIONS CAN BE DONE IN
* CLOSED FORM. TO SAVE TIME ONE CAN SET NOP = 1, LEAVING ONLY ONE "INTE-
* GRATION POINT". IN THIS CASE, ADJUSTMENTS IN THE VALUES OF THE TRIG
* FUNCTIONS ARE MADE AT STATEMENT 87 SO THAT THE INTEGRAL WILL COME OUT
* RIGHT. NOTE THAT IF NOP = 1, NOT IS MUST ALSO EQUAL 1 OR ELSE THE OFF-
* DIAGONAL MATRIX ELEMENTS MAY BE NONZERO AND THEREFORE INCORRECT.

READ IN INFORMATION WHICH REMAINS FIXED FOR MANY CASES.
RETURN TO STATEMENT 2 ONLY AFTER DUMPING RESULTS FOR THESE CASES.
A BLANK CARD HERE TERMINATES PROGRAM EXECUTION.

READ(5,100)NOT,S,NP2,NOP,IDIAG
IF(NOT.EQ.0)STOP
WRITE(6,130)
WRITE(6,123)
WRITE(6,101)PN,W
IF(NOT.GT.7)NOT = 7
IF(S.LE.0)S = 3
IF(S.GT.12)S = 12
ACC = 10.0*(1-S)
IF(NP2.LT.1)NP2 = 10
IF(NP2.GT.50)NP2 = 50
IF(NOP.GT.200)NOP = 200
IF(NOP.EQ.0)NOP = 50
IF(NOP.EQ.1)NOT = 1

PRINT THE FIXED INFORMATION READ FROM THE FIRST DATA CARD

WRITE(6,102)NOT
WRITE(6,104)S
WRITE(6,105)NP2
WRITE(6,108)NOP
WRITE(6,103)IDIAG
*****

SC = 1 IF COSINE TERMS ARE TO BE USED; 2 IF SINE TERMS ARE DESIRED
L = INDEX OF FIRST TERM IN FIELD EXPANSION
N = INDEX OF OVOID (1 = ELLIPSE, >30 = SQUARE)
R = ASPECT RATIO
B = ASSUMED VALUE OF THE BINDING CONSTANT
PMAW, PMIN = RANGE OF VALUES OF P2 TO BE SEARCHED
SZ = SEMIMINOR AXIS OF GUIDE

NOTE: NORMAL PROCEDURE IS TO LEAVE SZ BLANK, IN WHICH CASE SZ IS IGNORED.
IN THE PROGRAM B IS ALWAYS DEFINED IN TERMS OF AN AXIS OF UNITY. FOR A
GUIDE OF SIZE SZ, A NORMALIZED FREQUENCY B IS EQUIVALENT TO A NORMALIZED
FREQUENCY B*SZ IN A GUIDE OF SIZE 1.

ONE DATA CARD CORRESPONDS TO ONE INDIVIDUAL CASE

RETURN TO STATEMENT 1 IMMEDIATELY AFTER EACH CASE IS COMPLETE. A BLANK
CARD READ AT STATEMENT 1 CAUSES RESULTS FOR ALL CASES TO BE DUMPED
(PRINTED) AND CONTROL WILL THEN GO TO STATEMENT 2. A BLANK CARD THERE
TERMINATES THE JOB EXECUTION, SO TWO BLANK CARDS SHOULD FOLLOW THE LAST
DATA CARD.

```

```

C      J = 0
1 READ(5,121)SC,L,N,R,BP,PMAX,PMIN,SZ
  IF(R.LE.0.0)GO TO 999
C      IF(JDEX.GE.20)GO TO 999
      JDEX = JDEX + 1
      CALL SECOND(TO)
      B = BP
      IF(SZ.LE.0.0)GO TO 60
      B = BP*SZ
60 IF(PMAX.EQ.0.0)PMAX = 1.0
   IF(N.LE.0.0)N = 1.0
   IF(B.LE.0.0)B = 1.0
   IF(SC.LE.0)SC = 1
   IF(SC.GT.2)SC = 2
C      .....
C      DETERMINE S, THE INDEX OF THE LAST TERM IN THE FIELD EXPANSION
C      .....
      S = L + 2*(NOT-1)
      IF(R.NE.1.0)GO TO 65
      I = L/2
      I = 2*I
      IF(I.NE.L)GO TO 65
      S = L + 4*(NOT-1)
C      .....
C      PRINT INPUT DATA
C      .....
65 WRITE(6,111)
   WRITE(6,123)
   WRITE(6,111)
   WRITE(6,109).DEX,L,S,N,R,B
   IF(SZ.GT.0.0)WRITE(6,129)SZ,BP
   IF(PMAX.NE.1.0.OR.PMIN.NE.0.0)WRITE(6,110)PMAX,PMIN
C      .....
C      INITIALIZE STORAGE ARRAYS FOR THIS CASE
C      .....
DO 70 I = 6,13
DO 69 K = 1,NOT
69 CFF(JDEX,I-S,K) = 0.0
70 ROOT(JDEX,I) = 0.0
   ROOT(JDEX,3) = N
   ROOT(JDEX,4) = R
   ROOT(JDEX,5) = BP
   ROOT(JDEX,1) = 0
   ROOT(JDEX,2) = L
   ROOT(JDEX,3) = 5
C      .....
C      IDENTIFY SHAPE OF CROSS-SECTION
C      .....
      J = SC + 5
      I = 4
      IF(N.GT.1.0)GO TO 90
      IF(N-5)94,95,96
94 I = 9
   GO TO 92
95 I = 8
   GO TO 92

```

```

96 IF(NE.1.0)GO TO 92
1(
IF(LEQ.1.0)I = 1
90 IF(LE.30.)GO TO 92
I = 5
IF(LEQ.1.0)I = 3
92 CONTINUE
WRITE(6,112)PR(J),PR(I)
ROOT(JDEX,2) = PR(J)
ROOT(JDEX,1) = PR(I)
IF(NOP.EQ.1.AND.I.NE.1)WRITE(6,131)
.....
C
C
PN = NOP
W = PI/(PN*2.0)
IF(R.EQ.RL.AND.N.EQ.NL)GO TO 79
RL = R
NL = N
C
C
CALCULATE AND STORE RHO(PHI) AND D(RHO)/D(PHI)
DO 75 I = 1,NOP
PN = 2*(NOP - I) + 1
PHI = W*PN/2.0
CALL PERIM(N,R,PHI,RHO,DRDP)
STORE(I,1) = RHO
75 STORE(I,2) = DRDP
C
C
CALCULATE AND STORE COS(M*PHI) AND SIN(M*PHI) WHERE M CAN BE L.....S.
VALUES OF M ARE STORED AS INDX.
79 CONTINUE
IF(L.EQ.LLL.AND.S.EQ.LLS)GO TO 89
LLL = L
LLS = S
INDX(1) = L
IF(NOT.EQ.1)GO TO 86
LST = (S - L)/(NOT - 1)
DO 85 K = 2,NOT
INDX(K) = L + (K - 1)*LST
85 LST = 2*NOT + 2
IF(NOP.EQ.1)GO TO 87
DO 80 I = 1,NOP
PN = 2*(NOP - I) + 1
PHI = W*PN/2.0
DO 80 M = 3,LST,2
K = M/2
PN = INDX(K)
PN = PN*PHI
STORE(I,M) = COS(PN)
80 STORE(I,M+1) = SIN(PH)
GO TO 89
C
C
IF NOP = 1, ASSUME A CIRCLE. SET TRIG FUNCTIONS EQUAL TO CONSTANTS
C
CHOSEN SO THAT THE ANALYTICAL RESULTS FOR THE INTEGRAL ARE OBTAINED.
C
87 STORE(1,3) = SORT(.5)
STORE(1,4) = SORT(.5)
IF(INDX(1).NE.0)GO TO 89
STORE(1,3) = 1.0
.....
C
C
CONSTRUCT THE TABLE OF DETERMINANT VALUES FOR PMIN < P2 < PMAX.
C
FOR EACH VALUE OF P2, SUBROUTINE CHEBY CALCULATES THE MATRIX
C

```



```

C      GO, WHOSE DETERMINANT IS SOUGHT. SUBROUTINE MATINV CALCULATES
C      THE DETERMINANT.
C
      89 LST = NP2 + 1
      N = 3*HRLS
      IF(SC.EQ.2)W = 3*HLS
      IF(IDIAG.GT.0)WRITE(6,106)W
      PN = NP2
      PN = (PMAX - PMIN)/PN
      WW = PN/100.
      P2 = PMAX
      IF(PMAX.EQ.1.0)P2 = 1.0 - WW
      DO 200 M = 1,LST
      CALL CHEBY(NOP,SC)
      CALL MATINV(GD,NOT,C,DET,0)
      TABLE(M,1) = P2
      TABLE(M,2) = DET
      IF(IDIAG.GT.0)WRITE(6,107)M,P2,DET
      P2 = PMAX - M*PN
      200 IF(P2.LT.WW)P2 = WW
      IF(IDIAG.GT.0)WRITE(6,114)
C
      CALL SECOND(T1)
      T1 = T1 - TO
      WRITE(6,115)T1
C
C .....
C      MAKE TENTATIVE IDENTIFICATION OF ROOTS OF EQUATION DET(RLS) = 0 (OR T1S)
C
      NR = 0
      IF(TABLE(1,2).NE.0.0)GO TO 210
      NR = 1
      INDEX(NR) = 1001
      210 DO 220 M = 2,LST
      PN = ABS(TABLE(M,2))
      IF(PN.GT.0.0)GO TO 215
      NR = NR + 1
      INDEX(NR) = M + 1000
      GO TO 220
      215 PN = (TABLE(M,2)/PN)*TABLE(M-1,2)
      IF(PN.GE.0.0)GO TO 220
      NR = NR + 1
      INDEX(NR) = M - 1
      220 IF(NR.GE.20)GO TO 221
      221 WRITE(6,124)NR,JDEX
      IF(NR.EQ.0)GO TO 1
C
C .....
C      LOCATE THE ROOTS OF DET(RLS) PRECISELY.
C      LOOP DO 250 IS COMPLETED ONCE FOR EACH ROOT (I) IDENTIFIED ABOVE
C
      IF THERE ARE MORE THAN 9 ROOTS, MAKE NOTE OF IT, ONLY THE FIRST 9
      ROOTS WILL BE LISTED IN THE SUMMARY AT THE END. (IF THERE ARE MORE
      THAN 20 ROOTS, THE LAST ONES WILL HAVE BEEN MISSED COMPLETELY.)
C
      K = NR
      IF(NR.LE.8)GO TO 251
      K = K - 8
      WRITE(6,128)K
      K = 8
      251 IROOT(JDEX,1) = K
C
      DO 250 I = 1,NR

```

```

C      N = INDEX(I)
C      C4 RTT(TABLE,ACC,PN,NOP,SC,M,IO,IOIAG)
C      RTT RETURNS A FINAL MATRIX, GD, A SET OF COEFFICIENTS, C,
C      AND THE ROOT, P2. SEE THE SUBROUTINE FOR A DESCRIPTION OF PARAMETERS.
C      PRINT THE FINAL MATRIX, IF DESIRED: STORE ROOT: CALCULATE A FINAL VALUE
C      FOR THE DETERMINANT. ALSO PRINT THE COEFFICIENTS.
C      IF(I.LE.8)ROOT(JDEX,I+5) = P2
C      IF(IOIAG.LT.1)GO TO 253
C      WRITE(6,114)
C      DO 252 J = 1,NOT
C      252 WRITE(6,118)W,(GD(J,K),K=1,NOT)
C      253 WRITE(6,114)
C      CALL MATINV(GD,NOT,C,DET,O)
C      WRITE(6,125)DEX,I,P2,PN,IO,DET
C      PN = IH
C      IF(NOT.GT.1)WRITE(6,122)((PN,INDEX(K),C(K)),K=1,NOT)
C      WRITE(6,114)
C      IF(DET.EQ.0.0)WRITE(6,113)
C      IF(I.GT.8)GO TO 250
C      DO 249 J = 1,NOT
C      249 CFF(JDEX,I,J) = C(J)
C      250 CONTINUE
C      CALL SECOND(T1)
C      T1 = T1 - TQ
C      WRITE(6,126)T1
C      GO TO 1
C      .....
C      AFTER ALL CASES (20 OR LESS) HAVE BEEN COMPLETED, PRINT A TABLE
C      OF RESULTS GIVING THE ROOTS AND THE COEFFICIENT RATIOS FOR EACH CASE.
C      999 WRITE(6,123)
C      WRITE(6,123)
C      WRITE(6,123)
C      WRITE(6,116)
C      DO 900 I = 1,JDEX
C      K = IROOT(I,1) + 5
C      IF(IOIAG.LE.1)WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),
C      1(ROOT(I,J),J=2,K)
C      900 IF(IOIAG.GE.2)WRITE(6,127)I,ROOT(I,1),(IROOT(I,M),M=2,3),
C      1(ROOT(I,J),J=2,K)
C      IF(NOT.EQ.1)GO TO 2
C      IF(IOIAG.EQ.0)GO TO 2
C      WRITE(6,111)
C      WRITE(6,119)
C      DO 910 I = 1,JDEX
C      K = IROOT(I,1)
C      IF(K.EQ.0)GO TO 905
C      WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,5),
C      1(CFF(I,IO,1),IO=1,K)
C      DO 902 J = 2,NOT
C      902 WRITE(6,120)(CFF(I,IO,J),IO=1,K)
C      GO TO 910
C      905 WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,5)
C      910 CONTINUE
C      GO TO 2
C      END
C      SUBROUTINE RTT(TABLE,ACC,PN,NOP,SC,M,IO,IOIAG)

```

```

C SUBROUTINE RTT1 PRECISELY LOCATES THE ROOT SPECIFIED BY M, ID, AND THEN
C DETERMINES THE FIELD EXPANSION COEFFICIENTS C(J) CORRESPONDING TO
C THAT ROOT. ACC IS THE DESIRED ACCURACY OF THE ROOT, AND PN IS THE
C DIFFERENCE IN THE ACTUAL BOUNDS OBTAINED. ID IS THE NUMBER OF FUN-
C CTION EVALUATIONS NECESSARY. SC, IDIAG, AND NOP ARE AS IN MAIN. GO, THE
C FINAL MATRIX AT THE ROOT P2, IS ALSO RETURNED. (C, GD, AND P2 ARE ALL
C IN COMMON BLOCKS.)
C
C INTEGER SC
C DIMENSION STORE(200,16),INDX(7),GD(7,7),C(7),SINV(7,7)
C DIMENSION TABLE(51,2)
C COMMON/A/B,P2,P1,NOT
C COMMON/B/STORE,INDX,GD,C
C
C 700 FORMAT(/,'* SEARCH ID*,7X,*X0*,11X,*X1*,11X,*X2*,10X,*F0*,9X,*F1*,
C 19X,*F2*,11X,*X*,10X,*F*,10X,*DX*')
C 701 FORMAT(/,'* $$$$ ERROR IN HINTS OR MULLER, IER =*,13,* $$$$$$*')
C
C IF(IDIAG.GE.3)WRITE(6,700)
C K = 0
C ID = 0
C IF(M.LT.1000)GO TO 230
C P2 = TABLE(M-1000,1)
C CALL CHEBY(NOP,SC)
C ID = 1
C GO TO 248
C
C 230 G1 = TABLE(M+1,2)
C G2 = TABLE(M,2)
C X1 = TABLE(M+1,1)
C X2 = TABLE(M,1)
C DN = X2 - X1
C
C IF G1 AND G2 ARE TWO OR MORE ORDERS OF MAGNITUDE DIFFERENT, USE
C A HALF-INTERVAL SEARCH TO NARROW THE RANGE X1 TO X2
C
C 235 MW = -G1/G2
C IF(MW.LT.100..AND.MW.GT..01)GO TO 240
C IF(DN.LT.ACC)GO TO 240
C DN = DN/3.0
C
C CALL HINTS(X1,X2,G1,G2,DN,PN,NOP,SC,ID,IDIAG,IER)
C IF(IER.GT.0)WRITE(6,701)IER
C
C K = K + ID
C IF(PN.EQ.0.0)GO TO 248
C GO TO 235
C
C MULLER'S METHOD TO GET THE ROOT PRECISELY
C
C 240 CALL MULLER(X1,X2,G1,G2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C IF(IER.GT.0)WRITE(6,701)IER
C
C 248 ID = ID + K
C M = 1
C IF(NOT.EQ.1)GO TO 280
C
C CALCULATE THE COEFFICIENTS: SUBROUTINE LOAD COPIES (NOT-1) OF THE
C NOT ROWS OF GD INTO ARRAY SINV AND VECTOR C. SUBROUTINE MATINV SOLVES
C THE (NOT-1)X(NOT-1) SYSTEM FOR THE COEFFICIENTS AND RETURNS THEM IN C.
C
C CALL LOAD(GD,SINV,C,NOT,M)
C LST = NOT - 1

```

```

C      MATINV(SINV,LST,C,DN,1)
C      STACK THE COEFFICIENTS IN THE PROPER ORDER IN VECTOR C, INCLUDING 1.000
C
      IF(M.EQ.NOT)GO TO 280
      DO 270 J = M,LST
      K = NOT + M - J
      270 C(K) = C(K-1)
      280 C(M) = 1.0
      RETURN
      END
      SUBROUTINE LOAD(SS,SINV,C,NOT,ID)
C      SUBROUTINE LOAD LOADS SS (NEE GO) INTO ARRAY SINV, ELIMINATING
C      THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
C      IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
C      LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
C      INATED. THUS, SINV IS (NOT-1) BY (NOT-1) IN DIMENSION, AND C IS A VEC-
C      TOR (NOT-1) IN LENGTH.
C      NOTE THAT ONLY THE FIRST THREE DIAGONAL ELEMENTS ARE SEARCHED FOR
C      THE SMALLEST VALUE.
C
      DIMENSION SS(7,7),SINV(7,7),C(7)
      G = 1.0E+321
      ID = 1
      K = NOT
      IF(K.GT.3)K = 3
      DO 200 J = 1,K
      H = ABS(SS(J,J))
      IF(H.GT.G)GO TO 200
      G = H
      ID = J
      200 CONTINUE
C
      JJ = 0
      DO 300 J = 1,NOT
      IF(J.EQ.ID)GO TO 300
      JJ = JJ + 1
      C(JJ) = -SS(J,ID)
      KK = 0
      DO 299 K = 1,NOT
      IF(K.EQ.ID)GO TO 299
      SINV(JJ,KK) = SS(J,K)
      299 CONTINUE
      300 CONTINUE
      RETURN
      END
      SUBROUTINE MULLER(X0,X2,F0,F2,ACC,PH,NOP,SC,ID,DTAG,IER)
C      MULLER'S ALGORITHM FOR THE SOLUTION OF DET(RLS) = 0
C
      INVERSE PARABOLIC INTERPOLATION USING POINTS
      (X0,F0), (X1,F1), AND (X2,F2)
      TO FIND THE NEW POINT (X,F), WHERE X0 < X1 < X2 AND F0*F2 < 0
      ALWAYS HOLD. SIGN IS THE SIGN OF F0, AND DOES NOT CHANGE FROM ITS
      INITIAL VALUE. THE TWO ROOTS OF THE QUADRATIC NECESSARILY LIE
      BETWEEN X0 AND X1, AND BEYOND X0 OR X2 IF (A) SIGN*F1 < 0
      BETWEEN X1 AND X2, AND BEYOND X0 OR X2 IF (B) SIGN*F1 > 0
      (THAT IS, FOR EXAMPLE, IN CASE (A) THERE IS NO ROOT BETWEEN X1 AND X2.)
      CHOOSE THE ROOT INSIDE THE INTERVAL.
      XL IS THE LARGEST X FOR WHICH F*SIGN > 0
      XU IS THE SMALLEST X FOR WHICH F*SIGN < 0
      INTEGER SC

```

```

C      DIMENSION STORE(200,10),INDX(7),GD(7,7),C(7)
C      COMMON/8/STORE,INDX,GD,C
C
C      ID = 0
C      IER = 0
C      PN = X2 - X0
C      IF(X2.LE.X0)GO TO 290
C      W = ALOG(PN/ACC)/.693147
C      LST = W + 1.5
C
C      LST GIVES THE NUMBER OF ITERATIONS WHICH A HALF-INTERVAL SEARCH
C      WOULD REQUIRE FOR THESE INITIAL VALUES AND DESIRED ACCURACY. IF
C      MULLER'S ALGORITHM EXCEEDS THIS, EXIT WITH ERROR CONDITION 3.
C
C      700 FORMAT(* MULLR:*,I3.3F13.10,3E11.4,F13.10,2E11.4)
C      701 FORMAT(* MULLR:*,I3.3F13.10,3E11.4)
C
C      ASSUME THAT THE INITIAL VALUES ARE SUCH THAT X0 < X2
C      IF NOT, RETURN (IER = 4)
C
C      XL = X0
C      XU = X2
C      SIGN = F0/ABS(F0)
C
C      SPLIT THE INTERVAL AND EVALUATE THE DETERMINANT AT THE MIDPOINT
C
C      40 X = (XL + XU)/2.0
C      P2 = X
C      CALL CHEBY(NOP,SC)
C      CALL MATINV(GD,NOT,C,F,0)
C      ID = ID + 1
C      IF(F*SIGN)50,200,60
C      50 XU = X
C      GO TO 100
C      60 XL = X
C
C      EVALUATE THE NEW X FROM THE OLD X0, X1, AND X2
C
C      100 X1 = X
C      F1 = F
C      DD01 = (F1 - F0)/(X1 - X0)
C      DD12 = (F2 - F1)/(X2 - X1)
C      DD012 = (DD12 - DD01)/(X2 - X0)
C      W = DD01 + (X1 - X0)*DD012
C      U = DD012/W
C      GL = F1/W
C
C      U = 1.0 - 4.0*GL
C      IF(U.LT.0.0)GO TO 260
C      U = 2.0*GL/(1.0 + SORT(U))
C      X = X1 - U
C      IF(X.GT.X0.AND.X.LT.X2)GO TO 145
C      U = 1.0 - 4.0*GL*DD012/W
C      X = X1 - U
C      145 CONTINUE
C      IF(X.GT.1.0.OR.X.LT.0.0)GO TO 270
C      P2 = X
C      CALL CHEBY(NOP,SC)
C      ID = ID + 1
C      PN = ABS(XU - XL)
C      IF(PN.LT.ACC)GO TO 210
C      IF(ID.GT.LST)GO TO 280
C      CALL MATINV(GD,NOT,C,F,0)

```

```

C      IF (TAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X,F,PN
C
C      RESET THE UPPER OR LOWER BOUND ON THE ROOT
C
C      IF(F*SIGN)150,200,155
150 XU = X
    GO TO 160
155 XL = X
C
C      CHANGE X0 OR X2 TO THE PREVIOUS X1; THE NEW X1 IS X
C
160 IF(F1*SIGN)165,200,170
165 X2 = X1
    F2 = F1
    GO TO 175
170 X0 = X1
    F0 = F1
175 GL = ABS(U)
    IF((F*F1).LT.0.0)GO TO 100
    IF(GL.GT.ACC)GO TO 100
C
C      IF POINTS ACCUMULATE CLOSELY ON ONE SIDE OF THE TRUE ROOT, A SINGLE
C      SMALL JUMP MAY SPAN THE ROOT, THEREFORE CHANGE X BY A LITTLE LESS
C      THAN ACC AND EVALUATE THE DETERMINANT. IF THIS FAILS (THAT IS, IF THE
C      FUNCTION HAS THE SAME SIGN AGAIN) SPLIT THE INTERVAL IN HALF AND
C      CONTINUE.
C
C      LINEAR EXTRAPOLATION PAST THE ANTICIPATED ZERO
C      IF X = X1, DO THIS ANYWAY AND THEN EXIT.
C
      W = 1.0E-12
      IF(X.NE.X1)GO TO 180
      XT = W
      IF((F*SIGN).GT.0.0)XT = -W
      GO TO 185
180 XT = 2.0*F*(X1 - X)/(F1 - F)
      GL = ABS(XT)
      IF(GL.GT.ACC)GO TO 100
      IF(GL.GT.W)GO TO 185
      XT = W*XT/GL
185 XT = X - XT
      P2 = XT
      CALL CHEBY(NOP,SC)
      CALL MATINV(GO,NOT,C,FT,0)
      ID = ID + 1
      IF(IDIAG.GE.3)WRITE(6,701)ID,XT,FT
      IF((FT*F))190,200,40
190 IF(FT*SIGN)192,200,193
192 XU = XT
      X2 = XT
      F2 = FT
      IF(X-X1)100,205,100
193 XL = XT
      X0 = XT
      F0 = FT
      IF(X-X1)100,205,100
C
C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH MAY HAVE BEEN
C      ALTERED IN MATINV) IS RECOMPUTED.
C
200 PN = 0.0
    GO TO 209
C
C      IF X = X1, TAKE THIS AS THE ROOT
C

```

```

208 P2 X
PN( ABS(XU - XL)
C
209 CALL CHEBY(NOP,SC)
ID = ID + 1
C
C NORMAL RETURN: PN,LT,ACC
C
210 IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X
RETURN
C
C ERROR RETURN 1: COMPLEX ROOT
C
260 IER = 1
GO TO 299
C
C ERROR RETURN 2: PROCEDURE IS OSCILLATING
C
270 IER = 2
GO TO 299
C
C ERROR RETURN 3: MORE ITERATIONS THAN HALF-INTERVAL SEARCH WOULD REQUIRE
C
280 IER = 3
GO TO 210
C
C ERROR RETURN 4: X0 > X2 INITIALLY
C
290 IER = 4
299 ID = ID + 1
CALL CHEBY(NOP,SC)
RETURN
END
SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C
C SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE THE ROOTS OF
C THE EQUATION DET(RLS) = 0 OR DET(TLS) = 0 TO WITHIN A DESIRED ACCURACY
C ACC. ID IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
C
C INTEGER SC
C DIMENSION STORE(200,16),INDX(7),GD(7,7),C(7)
C COMMON/A/B,P2,P1,NOT
C COMMON/B/STORE,INDX,GD,C
C
700 FORMAT(= HINTS:*,I3,I3X,2F13.10,I1X,2E11.4,F13.10,2E11.4)
C
ID = 0
IER = 0
100 PN = ABS(X2 - X1)
C
IF(PN.LT.ACC)RETURN
ID = ID + 1
IF(ID.GT.25)GO TO 230
P2 = (X1 + X2)/2.0
CALL CHEBY(NOP,SC)
CALL MATINV(GD,NOT,C,G,0)
C
IF(IDIAG.GE.3)WRITE(6,700)ID,X1,X2,G1,G2,P2,G,PN
C
IF(G*(G1/ABS(G1)))120,210,130
120 X2 = P2
G2 = G
GO TO 100
130 X1 = P2
G1 = G
GO TO 100

```

```

C      (
C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE BEEN
C      ALTERED IN MATINV) IS RECOMPUTED.
C
210  PN = 0.0
      CALL CHEBY(NOP,SC)
      ID = ID + 1
      RETURN
C
C      ERROR RETURN: TOO MANY ITERATIONS
C
230  IER = 6
      RETURN
      END
C
      SUBROUTINE CHEBY(NOP,SC)
C
      INTEGER S,SC
      DIMENSION STORE(200,16),INDX(7),GD(7,7),C(7)
      DIMENSION BUST(29),BKST(29)
      COMMON/A/B,P2,PI,NOT
      COMMON/B/STORE,INDX,GD,C
      COMMON/C/P,OMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,L,S
C
      CHEBYSCHV INTEGRATION OF RLS(PHI) TO OBTAIN THE MATRIX ELEMENTS
C
      NMAX = INDX(NOT) + 1
      IF(NMAX.EQ.1)NMAX = 2
      OR = 0.0
      NN = 0
      P = SORT(P2)
      OMP = SORT(1.0 - P2)
      PTS = NOP
      W = PI/(PTS+2.0)
      DO 90 I = 1,NOT
      DO 90 J = 1,NOT
      90  GD(I,J) = 0.0
C
      LOOP DO 100 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
C
      DO 100 JUJ = 1,NOP
C
      OBTAIN RHO(PHI) AND D(RHO)/D(PHI)
C
      RH = STORE(JUJ,1)
      DR = STORE(JUJ,2)
C
      CALCULATE AND STORE THE J BESSEL FUNCTIONS WHICH WILL BE NEEDED
      JO IS STORED AS BUST(1), J1 IS STORED AS BUST(2), ETC.
C
      ARG = PI*B*RH+SQRT(1.0 - P2)
      CALL BESJ(ARG,OR,NMAX,BUST,NZ)
C
      CALCULATE AND STORE THE K BESSEL FUNCTIONS WHICH WILL BE NEEDED.
      KO WILL BE STORED AS BKST(1), K1 WILL BE STORED AS BKST(2), ETC.
C
      ARG = PI*B*RH+SQRT(P2)
      CALL BESKN(ARG,NN,1,NMAX,BKST,NZ)
C
      CALCULATE R(L,S) OR T(L,S) FOR ALL RELEVANT VALUES OF L AND S:
      LOOP DO 100 I... CORRESPONDS TO L
      LOOP DO 100 J... CORRESPONDS TO S
C
      DO 100 I = 1,NOT
      L = INDX(I)

```



```

C      OBTAIN COS(L*PHI) AND SIN(L*PHI)
C      CL = STORE(UUU,2*1+1)
C      SL = STORE(UUU,2*1+2)
C      RETRIEVE THE K BESSEL FUNCTIONS OF ORDER L AND L-1
C      BK = BKST(L+1)
C      LL = L - 1
C      IF(L.EQ.0) LL = 1
C      BKM = BKST(LL+1)
C      DO 100 J = 1,NOT
C      S = INDX(J)
C      OBTAIN COS(S*PHI) AND SIN(S*PHI)
C      CS = STORE(UUU,2*J+1)
C      SS = STORE(UUU,2*J+2)
C      OBTAIN THE J BESSEL FUNCTION OF ORDER S AND S-1
C      BJ = BJST(S+1)
C      K = S - 1
C      IF(S.EQ.0) K = 1
C      BJM = BJST(K+1)
C      IF(S.EQ.0) BJM = -BJM
C      100 GD(I,J) = GD(I,J) + W*RLS(SC)
C      RETURN
C      END
C      FUNCTION RLS(SC)
C      EVALUATE THE INTEGRAND OF THE LINE INTEGRAL, WHICH BECOMES THE
C      MATRIX ELEMENT R(L,S) (COSINE EXPANSION) OR T(L,S) (SINE EXPANSION)
C      INTEGER S,SC
C      COMMON/A,B,P2,P1,NOT
C      COMMON/C/P,OMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,L,S
C      XS = S
C      XL = L
C      EPS = 1.0
C      IF(L.GT.0) EPS = 2.0
C      TERM = BK*BJM*OMP + P*BJ*BKM
C      TERM = 2.0*B*RH*TERM
C      IF(L.EQ.S) GO TO 90
C      DO L = XL,S
C      TERM = TERM + 2.0*DD*BK*BJ/PI
C      90 F = EPS*TERM
C      NOTE THAT I=-(L+1) IS NOT INCLUDED IN F
C      E = 0.0
C      IF(L.EQ.S) GO TO 100
C      E = 2.0*EPS*BK*BJ/DR/(RH*PI)
C      NOTE THAT I=-(L+1) IS NOT INCLUDED IN E
C      100 CONTINUE
C      IF(SC.EQ.2) GO TO 110
C      TERM = XS*CL*SS - XL*SL*CS

```

```

      RI = 4.0*(F*CL*CS + E*TERM)
      RETURN
C
110 TERM = XS*SL*CS - XL*SS*CL
    RLS = 4.0*(F*SL*SS - E*TERM)
    RETURN
    END
SUBROUTINE PERIM(N,R,PHI,RHO,DRDP)
C
C  EVALUATE RHO(PHI) AND D(RHO)/D(PHI) ON THE PERIMETER
C
C  REAL N
C
C  IF(R.NE.1.0.OR.N.NE.1.0)GO TO 100
    RHO = 1.0
    DRDP = 0.0
    RETURN
C
C 100 CONTINUE
    CP = COS(PHI)
    SP = SIN(PHI)
    PN = 2.0*N
    DD = (CP/R)*PN + SP*PN
    PN = 1.0/PN
    RHO = 1.0/(DD*PN)
C
C  PN = 2.0*N - 2.0
    AA = SP*PN - ((CP/R)*PN)/(R*R)
    DRDP = -RHO*SP*CP*AA/DD
    RETURN
    END
SUBROUTINE MATINV(A,N,B,DETER,M)
C
C  DIMENSION A(7,7),B(7,1),INDEX(7,3)
C 10 DETER = 1.0
C 15 DO 20 J = 1,N
C 20 INDEX(J,3) = 0
C 30 DO 50 I = 1,N
C
C  SEARCH FOR PIVOT ELEMENT
C
C 40 AMAX = 0.
C 45 DO 105 J = 1,N
    IF (INDEX(J,3)-1) 60,105,60
C 60 DO 100 K = 1,N
    IF (INDEX(K,3)-1) 80,100,740
C 80 IF(AMAX-ABS(A(J,K))) 85,100,100
C 85 IROW = J
C 90 ICOL = K
    AMAX = ABS(A(J,K))
C 100 CONTINUE
C 105 CONTINUE
    IF(AMAX.EQ.0.0)GO TO 750
    INDEX(ICOL,3) = INDEX(ICOL,3) + 1
C 260 INDEX(I,1) = IROW
C 270 INDEX(I,2) = ICOL
C
C  INTERCHANGE ROWS TO PUT PIVOT ELEM.IN DIAG.
C
C 130 IF(IROW=ICOL) 140,310,140
C 140 DETER = -DETER
C 150 DO 200 L = 1,N
C 160 SWAP = A(IROW,L)
C 170 A(IROW,L) = A(ICOL,L)
C 200 A(ICOL,L) = SWAP

```

```

210 DO 50 L = 1, M
220 SW = B(IROW, L)
230 B(IJROW, L) = B(ICOLUMN, L)
250 B(ICOLUMN, L) = SWAP
C
C   DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
310 PIVOT = A(ICOLUMN, ICOLUMN)
   DETER = DETER * PIVOT
330 A(ICOLUMN, ICOLUMN) = 1.0
340 DO 350 L = 1, N
350 A(ICOLUMN, L) = A(ICOLUMN, L) / PIVOT
355 IF (M) 380, 380, 360
360 DO 370 L = 1, M
370 B(ICOLUMN, L) = B(ICOLUMN, L) / PIVOT
C
C   REDUCE NON PIVOTS ROWS
C
380 DO 550 L1 = 1, N
390 IF (L1 - ICOLUMN) 400, 550, 400
400 T = A(L1, ICOLUMN)
420 A(L1, ICOLUMN) = 0.
430 DO 450 L = 1, N
450 A(L1, L) = A(L1, L) - A(ICOLUMN, L) * T
455 IF (M) 550, 550, 460
460 DO 500 L = 1, M
500 B(L1, L) = B(L1, L) - B(ICOLUMN, L) * T
550 CONTINUE
C
C   INTERCHANGE COLUMNS
C
600 DO 710 I = 1, N
610 L = N + 1 - I
620 IF (INDEX(L, 1) - INDEX(L, 2)) 630, 710, 630
630 JROW = INDEX(L, 1)
640 JCOLUMN = INDEX(L, 2)
650 DO 705 K = 1, N
660 SWAP = A(K, JROW)
670 A(K, JROW) = A(K, JCOLUMN)
700 A(K, JCOLUMN) = SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
C
750 DETER = 0.0
   RETURN
   END

```

Program Listing

- IDGS -

(Reference: Vol. I, Section II.3.3)


```

      'F' MAX.NE.1.0.0R.PMIN.NE.0.0)WRITE(6,110)PMAX.PMIN
      (

```

```

      .....
      INITIALIZE STORAGE ARRAYS FOR THIS CASE
      .....

```

```

      DO 70 I = 7,14
      DO 69 K = 1,NOT
      CFS(JDEX,I-6,K) = 0.0
      CFA(JDEX,I-6,K) = 0.0
      ROOS(JDEX,I-6) = 0.0
      69 ROOT(JDEX,I) = 0.0
      70 ROOT(JDEX,3) = N
      ROOT(JDEX,4) = R
      ROOT(JDEX,5) = D
      ROOT(JDEX,6) = B
      IROOT(JDEX,1) = 0
      IROOT(JDEX,2) = SC - 1
      IROOT(JDEX,3) = NOT + SC - 2
      IROOT(JDEX,4) = 0

```

```

      .....
      IDENTIFY CROSS-SECTION FOR THIS CASE
      .....

```

```

      J = SC + 5
      I = 4
      IF(N.GT.1.0)GO TO 90
      IF(N-0.5)94,95,96
      94 I = 9
      GO TO 92
      95 I = 8
      GO TO 92
      96 IF(N.NE.1.0)GO TO 92
      I = 2
      IF(R.EQ.1.0)I = 1
      90 IF(N.LT.30.)GO TO 92
      I = 5
      IF(R.EQ.1.0)I = 3
      92 CONTINUE
      WRITE(6,109)PR(J),PR(1)
      ROOT(JDEX,2) = PR(J)
      ROOT(JDEX,1) = PR(1)
      IF(NOP.EQ.1.AND.1.NE.1)WRITE(6,131)

```

```

      .....
      FIND LST. THE CODE FOR DETERMINING WHEN TO REPEAT CALCULATIONS
      LST = 1: SKIP INDEX AND PERIMETER CALCULATIONS
      LST = 2: RECALCULATE INDICES
      LST = 3: RECALCULATE PERIMETER
      LST = 4: RECALCULATE BOTH INDICES AND PERIMETER

```

```

      CALL CODE(N,R,NL,RL,SC,LSC,LST)
      IF(LST.LT.3)GO TO 80
      CALCULATE AND STORE RHO(PHI) AS STORE(I,1), I = 1,NOP
      CALCULATE AND STORE D(RHO)/D(PHI) AS STORE(I,2), I = 1,NOP
      CALCULATE AND STORE COS(M*PHI) AND SIN(M*PHI) AS STORE(I,2*M+1)
      AND STORE(I,2*M+2), RESPECTIVELY.

```

```

      PN = NOP
      W = PI/(PN*2.0)
      DO 79 I = 1,NOP
      PN = 2*(NOP - I) + 1
      PHI = W*PN/2.0

```

```

CALL PERIM(N,R,PHI,RHO,DRDP)
S1 = (1.1) = RHO
S1...E1(1.2) = DRDP
IF(JDEX.GT.1)GO TO 79
DO 75 M = 1,NT2
PN = M
PN = PN*PHI
STORE(1.2*M+1) = COS(PN)
75 STORE(1.2*M+2) = SIN(PN)
79 CONTINUE
IF(NOP.GT.1)GO TO 80
C
C IF NOP = 1, ASSUME A CIRCLE. SET TRIG FUNCTIONS EQUAL TO CONSTANTS
C CHOSEN SO THAT THE ANALYTICAL RESULTS FOR THE INTEGRALS ARE OBTAINED
C
DO 87 M = 1,NT2
STORE(1.2*M+1) = SORT(.5)
87 STORE(1.2*M+2) = SORT(.5)
C
80 IF(LST.EQ.1.OR.LST.EQ.3)GO TO 89
C
C DETERMINE THE INDEX PAIRS (L,S) OF ALL THE MATRIX ELEMENTS NEEDED
C BY CALLING SUBROUTINE REL.
C
CALL REL(N,R,SC,IDIAG)
C
C .....
C
C CONSTRUCT TABLE OF VALUES OF DET(SYM) AND DET(ASY) VS P2. GIVEN B.
C TO SAVE TIME (SINCE R1(L,S) AND R2(L,S) ARE INDEPENDENT OF D) THIS
C IS DONE FOR EACH OF THE VALUES OF D READ INTO THE ARRAY DST WHEN THE
C COUNTER, ND, IS EQUAL TO 1. THIS SECTION IS SKIPPED WHEN ND.GT.1.
C
89 LST = NP2 + 1
IF(ND.GT.1)GO TO 205
IF(IDIAG.GT.0)WRITE(6,106)
PN = NP2
PN = (PNAX - PMIN)/PN
M = PN/100.
P2 = PMAX
IF(PMAX.EQ.1.0)P2 = 1.0 - W
IF(NMAX.GT.1.AND.IDIAG.GT.0)WRITE(6,132)((DST(I),DST(I)),I=1,NMAX)
NR = 2*NMAX + 1
DO 200 M = 1,LST
TABLE(M,1) = P2
DO 190 I = 1,NMAX
D = DST(I)
CALL MATR(NOP,SC,I)
CALL MATINV(SYM,NOT.C,DET,0)
TABLE(M,2+I) = DET
CALL MATINV(ASY,NOT.C,XXX,0)
190 TABLE(M,2+I+1) = XXX
IF(IDIAG.GT.0)WRITE(6,107)M,(TABLE(M,I),I=1,NR)
P2 = PMAX - M*PN
200 IF(P2.LT.W)P2 = W
IF(IDIAG.GT.0)WRITE(6,114)
D = DST(1)
C
CALL SECOND(T1)
T1 = T1 - T0
WRITE(6,115)T1
GO TO 209
C
C WHEN ND.GT.1, RECOVER THE TABLE OF VALUES OF DET(SYM) AND DET(ASY)
C VS P2 CALCULATED EARLIER.
C

```



```

XV 3M R2
DO 256 J = 1,NT2
  WRITE(6,118)XXX,(R2(J,K),K=1,NOT)
  WRITE(6,114)
  XXX = 3HALP
DO 257 J = 1,NOT
  WRITE(6,118)XXX,(ALPHA(J,K),K=1,NT2)
  WRITE(6,114)
  XXX = 3HR*A
DO 260 J = 1,NOT
  DO 259 K = 1,NOT
    M = 0
    DO 258 LST = 1,NT2
      ON = ALPHA(J,LST)*R2(LST,K)
      IF(ON.EQ.0.0)GO TO 256
      M = M + 1
      CFS(M,8,10) = DN
    258 CONTINUE
    259 WRITE(6,118)XXX,(CFS(LST,8,10),LST=1,M)
    260 WRITE(6,114)
    IF(1DIAG.LT.5)GO TO 265
    XXX = 3M*KN
    J = NOT + NT2 + 1
  WRITE(6,118)XXX,(BST(K),K=1,J)
  WRITE(6,114)
  .....
C C C C C
C PRINT THE FINAL MATRIX, IF DESIRED: STORE ROOT: CALCULATE A FINAL VALUE
C FOR THE DETERMINANT. ALSO PRINT THE COEFFICIENTS.
C
265 IF(1.LE.8)ROOT(JDEX,1+6) = P2
  IF(1DIAG.LT.1)GO TO 253
  XXX = 3MSYM
  WRITE(6,114)
  DO 252 J = 1,NOT
    252 WRITE(6,118)XXX,(SYM(J,K),K=1,NOT)
    253 WRITE(6,114)
    CALL MATINV(SYM,NOT,C,DET,0)
    WRITE(6,125)XXX,JDEX,1,P2,PM,1D,DET
    IF(NOT.GT.1)WRITE(6,122)(C(K),K=1,NOT)
    WRITE(6,114)
    IF(DET.EQ.0.0)WRITE(6,113)
    IF(1.GT.8)GO TO 250
    DO 249 J = 1,NOT
      249 CFS(JDEX,1,J) = C(J)
    250 CONTINUE
  C C C C C
  .....
C 300 CONTINUE
  IF(NR.GT.0)GO TO 321
  WRITE(6,123)
  IF(IND-NMAX)6,1,1
  321 K = NR
  IF(NR.LE.8)GO TO 351
  K = K - 8
  WRITE(6,128)K
  K = B
  351 IROOT(JDEX,4) = K
  C
  DO 350 I = 1,NR
    M = INDEX(I,2)
    C

```

```

C      CALL RTT(TABLE,ACC,PN,NOP,SC,3,M,ID,IDIAG)
C      R(      RETURNS A FINAL MATRIX, ASY, A SET OF COEFFICIENTS, d, ND
C      THE ROOT, P2. SEE THE SUBROUTINE FOR A DESCRIPTION OF PARAMETERS.
C
C      PRINT THE FINAL MATRIX, IF DESIRED: STORE ROOT: CALCULATE A FINAL VALUE
C      FOR THE DETERMINANT. ALSO PRINT THE COEFFICIENTS.
C
      IF(1.LE.8)ROOTS(JDEX,1) = P2
      IF(IDIAG.LT.1)GO TO 353
      XXX = 3MAS,
      WRITE(6,114)
      DO 352 J = 1,NOT
      352 WRITE(6,118)XXX,(ASY(J,K),K=1,NOT)
      353 WRITE(6,114)
      CALL MATINV(ASY,NOT,C,DET,0)
      WRITE(6,125)XXX,JDEX,1,P2,PN,ID,DET
      IF(NOT.GT.1)WRITE(6,122)(C(K),K=1,NOT)
      WRITE(6,114)
      IF(DET.EQ.0.0)WRITE(6,113)
      IF(1.GT.8)GO TO 350
      DO 349 J = 1,NOT
      349 CFA(JDEX,1,J) = C(J)
      350 CONTINUE
      .....
C      CALL SECOND(T1)
C      T1 = T1 - TO
C      WRITE(6,126)T1
C      IF(ND-NMAX)5,1,1
C
C      RETURN TO EITHER PROCESS ANOTHER VALUE OF D FROM THE LAST DATA CARD,
C      OR TO READ A NEW CARD.
C
C      .....
C
C      AFTER ALL CASES (20 OR LESS) HAVE BEEN COMPLETED, PRINT A TABLE
C      OF RESULTS GIVING THE ROOTS AND THE COEFFICIENT RATIOS FOR EACH CASE.
C
C      SYMMETRIC ROOTS:
C
      999 WRITE(6,123)
      WRITE(6,123)
      WRITE(6,123)
      XXX = 4M
      WRITE(6,116)XXX
      DO 900 I = 1,JDEX
      MAX = IROOT(1,1) + 6
      IF(IDIAG.LE.1)WRITE(6,117)I,ROOT(1,1),(IROOT(1,M),M=2,3),
      1(ROOT(1,J),J=2,MAX)
      900 IF(IDIAG.GE.2)WRITE(6,127)I,ROOT(1,1),(IROOT(1,M),M=2,3),
      1(ROOT(1,J),J=2,MAX)
C
      IF(NOT.EQ.1)GO TO 911
      IF(IDIAG.EQ.0)GO TO 911
      WRITE(6,111)
      WRITE(6,119)
      DO 910 I = 1,JDEX
      MAX = IROOT(1,1)
      IF(MAX.EQ.0)GO TO 905
      WRITE(6,117)I,ROOT(1,1),(IROOT(1,M),M=2,3),(ROOT(1,J),J=2,6),
      1(CFS(1,K,1),K=1,MAX)
      DO 902 K = 2,NOT

```

```

902 WRITE(6,120)(CFS(I,M,K),(I,M,K),M=1,MAX)
DO 704 K = 1,NOT
  C 03 M = 1,MAX
903 C(M) = CFS(I,M,K)*(-1.0)**(K+1)
904 WRITE(6,120)(C(M),M=1,MAX)
GO TO 910
905 WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6)
910 CONTINUE
C
C ANTISYMMETRIC ROOTS:
C
911 WRITE(6,111)
XXX = 4HANTI
WRITE(6,116)XXX
DO 920 I = 1,JDEX
  MAX = IROOT(I,4)
  IF(MAX.EQ.0)GO TO 915
  IF(IDIAG.LE.1)WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),
  1(ROOT(I,J),J=2,6),(ROOTS(I,K),K=1,MAX)
  IF(IDIAG.GE.2)WRITE(6,127)I,ROOT(I,1),(IROOT(I,M),M=2,3),
  1(ROOT(I,J),J=2,6),(ROOTS(I,K),K=1,MAX)
GO TO 920
915 WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6)
920 CONTINUE
  IF(NOT.EQ.1)GO TO 2
  IF(IDIAG.EQ.0)GO TO 2
  WRITE(6,111)
  WRITE(6,119)
  DO 930 I = 1,JDEX
    MAX = IROOT(I,4)
    IF(MAX.EQ.0)GO TO 925
    WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6),
    1(CFAL(M,1),M=1,MAX)
    DO 922 K = 2,NOT
      WRITE(6,120)(CFA(I,M,K),M=1,MAX)
      DO 924 K = 1,NOT
        DO 923 M = 1,MAX
          C(M) = CFA(I,M,K)*(-1.0)**(K)
924 WRITE(6,120)(C(M),M=1,MAX)
GO TO 930
925 WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6)
930 CONTINUE
GO TO 2
END
SUBROUTINE CODE(N,R,NL,RL,SC,LSC,LST)
C
C REAL N,NL
C INTEGER SC
C
C FIND LST: CODE FOR DECIDING WHEN TO RECALCULATE INDICES (2).
C PERIMETER (3), OR BOTH (4).
C
C LST = 1
C IF(R.EQ.RL)GO TO 71
C
C FIND LST FOR CASES WITH R.NE.RL
C
C LST = 3
C IF(R.NE.1.0.AND.RL.NE.1.0)GO TO 73
C GO TO 72
C
C 71 IF(N.EQ.NL)GO TO 73
C
C FIND LST FOR CASES WITH R.EQ.RL
C
C LST = 3

```



```

C      MUJ _R'S METHOD TO GET THE ROOT PRECISELY
C
C      240 CALL MULLER(X1,X2,G1,G2,ACC,PN,NOP,SC,JJ,ID,IDIAG,IER)
C          IF(IER.GT.0)WRITE(6,701)IER
C
C      248 ID = ID + K
C          M = 1
C          IF(NOT.EQ.1)GO TO 280
C
C      CALCULATE THE COEFFICIENTS: SUBROUTINE LOAD COPIES (NOT-1) OF THE NOT
C      ROWS OF SYM OR ASY INTO ARRAY SINV AND VECTOR C. SUBROUTINE MATINV
C      SOLVES THE (NOT-1)X(NOT-1) SYSTEM FOR THE COEFFICIENTS AND RETURNS THEM
C      IN C
C
C      IF(JJ.EQ.3)GO TO 260
C      CALL LOAD(SYM,SINV,C,NOT,M)
C      GO TO 266
C
C      260 CONTINUE
C      CALL LOAD(ASY,SINV,C,NOT,M)
C      266 LST = NOT - 1
C      CALL MATINV(SINV,LST,C,DN,1)
C
C      STACK THE COEFFICIENTS IN THE PROPER ORDER IN VECTOR C. INCLUDING 1.000
C
C      IF(M.EQ.NOT)GO TO 280
C      DO 270 J = M,LST
C          K = NOT + M - J
C      270 C(K) = C(M-1)
C      280 C(M) = 1.0
C      RETURN
C      END
C
C      SUBROUTINE MULLER(X0,X2,F0,F2,ACC,PN,NOP,SC,JJ,ID,IDIAG,IER)
C
C      MULLER'S ALGORITHM FOR THE SOLUTION OF DET(RLS) = 0
C
C      INVERSE PARABOLIC INTERPOLATION USING POINTS
C      (X0,F0), (X1,F1), AND (X2,F2)
C      TO FIND THE NEW POINT (X,F), WHERE X0 < X1 < X2 AND F0*F2 < 0
C      ALWAYS HOLD. SIGN IS THE SIGN OF F0, AND DOES NOT CHANGE FROM ITS
C      INITIAL VALUE. THE TWO ROOTS OF THE QUADRATIC NECESSARILY LIE
C      BETWEEN X0 AND X1, AND BEYOND X0 OR X2 IF (A) SIGN*F1 < 0
C      BETWEEN X1 AND X2, AND BEYOND X0 OR X2 IF (B) SIGN*F1 > 0
C      (THAT IS, FOR EXAMPLE, IN CASE (A) THERE IS NO ROOT BETWEEN X1 AND X2.)
C      CHOOSE THE ROOT INSIDE THE INTERVAL.
C      X1 IS THE LARGEST X FOR WHICH F*SIGN > 0
C      X0 IS THE SMALLEST X FOR WHICH F*SIGN < 0
C
C      INTEGER SC
C      DIMENSION SYM(10,10),ASY(10,10),C(10)
C      COMMON/A/B,P2,P1,D,NOT,NT2
C      COMMON/E/SYM,ASY,C
C
C      ID = 0
C      IER = 0
C      PN = X2 - X0
C      IF(X2.LE.X0)GO TO 290
C      W = ALOG(PN/ACC)/.693147
C      LST = W + 1.5
C
C      LST GIVES THE NUMBER OF ITERATIONS WHICH A HALF-INTERVAL SEARCH
C      WOULD REQUIRE FOR THESE INITIAL VALUES AND DESIRED ACCURACY. IF
C      MULLER'S ALGORITHM EXCEEDS THIS, EXIT WITH ERROR CONDITION 3.
C
C

```



```

700 FQ AT(0,MULLR:0.13,3F13,10.3E11,4,F13,10.2E11,4)
701 FQ AT(0,MULLR:0.13,72X,F13,10.2E11,4)
C
C ASSUME THAT THE INITIAL VALUES ARE SUCH THAT X0 < X2
C IF NOT, RETURN (IER = 4)
C
C XL = X0
C XU = X2
C SIGN = F0/ABS(F0)
C
C SPLIT THE INTERVAL AND EVALUATE THE DETERMINANT AT THE MIDPOINT
C
40 X = (XL + XU)/2.0
P2 = X
CALL MATR(NOP,SC,1)
IF(JJ.EQ.2)CALL MATINV(SYM,NOT,C,F,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NOT,C,F,0)
ID = ID + 1
IF(F*SIGN)50,200,60
50 XU = X
GO TO 100
60 XL = X
C
C EVALUATE THE NEW X FROM THE OLD X0, X1, AND X2
C
100 X1 = X
F1 = F
DD01 = (F1 - F0)/(X1 - X0)
DD12 = (F2 - F1)/(X2 - X1)
DD012 = (DD12 - DD01)/(X2 - X0)
W = DD01 + (X1 - X0)*DD012
U = DD012/W
GL = F1/W
U = 1.0 - 4.0*U*GL
IF(U.LT.0.0)GO TO 260
U = 2.0*GL/(1.0 + SORT(U))
X = X1 - U
IF(X.GT.X0.AND.X.LT.X2)GO TO 145
U = 1.0 - 4.0*GL*DD012/W
U = 2.0*GL/(1.0 - SORT(U))
X = X1 - U
145 CONTINUE
IF(X.GT.1.0.OR.X.LT.0.0)GO TO 270
P2 = X
CALL MATR(NOP,SC,1)
ID = ID + 1
PN = ABS(XU - XL)
IF(PN.LT.ACC)GO TO 210
IF(ID.GT.LST)GO TO 280
IF(JJ.EQ.2)CALL MATINV(SYM,NOT,C,F,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NOT,C,F,0)
C
C IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X,F,PN
C
C RESET THE UPPER OR LOWER BOUND ON THE ROOT
C
IF(F*SIGN)150,200,155
150 XU = X
GO TO 160
155 XL = X
C
C CHANGE X0 OR X2 TO THE PREVIOUS X1; THE NEW X1 IS X
C
160 IF(F1*SIGN)165,200,170
165 X2 = X1

```

```

F2 = F1
GO 3 175
170 AC = X1
F0 = F1
175 GL = ABS(U)
IF(((F-F1).LT.0.0)GO TO 100
IF(GL.GT.ACC)GO TO 100
C
C IF POINTS ACCUMULATE CLOSELY ON ONE SIDE OF THE TRUE ROOT, A SINGLE
C SMALL JUMP MAY SPAN THE ROOT, THEREFORE CHANGE X BY A LITTLE LESS
C THAN ACC AND EVALUATE THE DETERMINANT. IF THIS FAILS (THAT IS, IF THE
C FUNCTION HAS THE SAME SIGN AGAIN) SPLIT THE INTERVAL IN HALF AND
C CONTINUE.
C
C LINEAR EXTRAPOLATION PAST THE ANTICIPATED ZERO
C
W = 1.0E-12
IF(X.NE.X1)GO TO 180
XT = W
IF((F-SIGN).GT.0.0)XT = -W
GO TO 185
180 XT = 2.0*F*(X1 - X)/(F1 - F)
GL = ABS(XT)
IF(GL.GT.ACC)GO TO 100
IF(GL.GT.W)GO TO 185
XT = W*XT/GL
185 XT = X - XT
P2 = XT
CALL MATR(NOP,SC.1)
IF(JJ.EQ.2)CALL MATINV(SW,NOT.C,FT,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NOT.C,FT,0)
ID = ID + 1
IF(IDIAG.GE.3)WRITE(6,701)ID,XT,FT
190 IF(FT.F)190,200,40
192 XU = XT
X2 = XT
F2 = FT
IF(X-X1)100,205,100
193 XL = XT
X0 = XT
F0 = FT
IF(X-X1)100,205,100
C
C IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH MAY HAVE BEEN
C ALTERED IN MATINV) IS RECOMPUTED.
C
200 PN = 0.0
GO TO 209
C
C IF X = X1, TAKE THIS AS THE ROOT
C
205 P2 = X
PN = ABS(XU - XL)
C
C 209 CALL MATR(NOP,SC.1)
ID = ID + 1
C
C NORMAL RETURN: PN,LT,ACC
C
210 IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X
RETURN
C
C ERROR RETURN 1: COMPLEX ROOT

```

```

C 260 IF = 1
C 261 GO TO 269
C 262 ERROR RETURN 2: PROCEDURE IS OSCILLATING
C 263
C 270 IER = 2
C 271 GO TO 299
C 272 ERROR RETURN 3: MORE ITERATIONS THAN HALF-INTERVAL SEARCH WOULD REQUIRE
C 273
C 280 IER = 3
C 281 GO TO 210
C 282 ERROR RETURN 4: X0 > X2 INITIALLY
C 283
C 290 IER = 4
C 291 ID = ID + 1
C 292 CALL MATR(NQP,SC,1)
C 293 RETURN
C 294 END
C 295 SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,JJ,ID,IDIAG,IER)
C 296
C 297 SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE THE ROOTS OF
C 298 THE EQUATION DET(RLS) = 0 OR DET(TLS) = 0 TO WITHIN A DESIRED ACCURACY
C 299 ACC. ID IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
C 300
C 301 INTEGER SC
C 302 DIMENSION SYM(10,10),ASY(10,10),C(10)
C 303 COMMON/A/B,P2,P1,D,NOT,NT2
C 304 COMMON/E/SYM,ASY,C
C 305
C 306 FORMAT(= HINTS:,,I3,I3X,2F13.10,11X,2E11.4,F13.10,2E11.4)
C 307
C 308 ID = 0
C 309 IER = 0
C 310 PN = ABS(X2 - X1)
C 311
C 312 IF(PN.LT.ACC)RETURN
C 313 ID = ID + 1
C 314 IF(ID.GT.25)GO TO 230
C 315 P2 = (X1 + X2)/2.0
C 316 CALL MATR(NQP,SC,1)
C 317 IF(JJ.EQ.2)CALL MATINV(SYM,NOT,C,G,0)
C 318 IF(JJ.EQ.3)CALL MATINV(ASY,NOT,C,G,0)
C 319
C 320 IF(IDIAG.GE.3)WRITE(6,700)ID,X1,X2,G1,G2,P2,G,PN
C 321
C 322 IF(G*(G1/ABS(G1)))120,210,130
C 323 120 X2 = P2
C 324 G2 = G
C 325 GO TO 100
C 326 130 X1 = P2
C 327 G1 = G
C 328 GO TO 100
C 329
C 330 IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE BEEN
C 331 ALTERED IN MATINV) IS RECOMPUTED.
C 332
C 333 210 PN = 0.0
C 334 CALL MATR(NQP,SC,1)
C 335 ID = ID + 1
C 336 RETURN
C 337 ERROR RETURN: TOO MANY ITERATIONS
C 338

```

```

230 IER = 6
240 EK = 3N
SUBROUTINE LOAD(SS,SINV,C,NOT,ID)
C
C SUBROUTINE LOAD LOADS SS (NEE SYM OR ASY) INTO ARRAY SINV, ELIMINATING
C THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
C IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
C LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
C INATED. THUS, SINV IS (NOT-1) BY (NOT-1) IN DIMENSION, AND C IS A VEC-
C TOR (NOT-1) IN LENGTH.
C
C NOTE THAT ONLY THE FIRST FOUR DIAGONAL ELEMENTS ARE SEARCHED FOR
C THE SMALLEST VALUE.
C
C DIMENSION SS(10,10),SINV(10,10),C(10)
G = 1.0E+321
ID = 1
K = NOT
IF(K.GT.4)K = 4
DO 200 J = 1,K
H = ABS(SS(J,J))
IF(H.GT.G)GO TO 200
G = H
ID = J
200 CONTINUE
C
JJ = 0
DO 300 I = 1,NOT
IF(J.EQ.ID)GO TO 300
JJ = JJ + 1
C(JJ) = -SS(J,ID)
KK = 0
DO 299 K = 1,NOT
IF(K.EQ.ID)GO TO 299
KK = KK + 1
SINV(JJ,KK) = SS(J,K)
299 CONTINUE
300 CONTINUE
RETURN
END
SUBROUTINE REL(N,R,SC,IDIAG)
C
C SUBROUTINE REL IDENTIFIES THOSE MATRIX ELEMENTS R(L,S) (COSINE EX-
C PANSION, SC = 1) OR T(L,S) (SINE EXPANSION, SC = 2) WHICH ARE NONZERO.
C THE VALUES OF THE ROW INDICES, L, ARE STORED IN VECTOR ID(I), AND
C THE VALUES OF THE INDICES, S, NBR(I) IN NUMBER, ARE THE ROWS OF
C MATRIX IDMAT(I,J). FOR EXAMPLE, FOR A FIVE-TERM COSINE EXPANSION
C WITH R = 1, THE NONZERO ELEMENTS ARE R(0,0), R(0,4)
C R(1,1), R(1,3)
C R(2,2)
C R(3,1), R(3,3)
C R(4,0), R(4,4)
C
C AND ID(I) = 0 IDMAT(I,J) = 0, 4 NBR(I) = 2
C 1 1, 3
C 2 2 1
C 3 1, 3 2
C 4 0, 4 2
C
C REAL N
C INTEGER SC
C DIMENSION STORE(200,42),IDMAT(20,10),ID(20),NBR(20)
C DIMENSION R1(10,10),R2(20,10)
C COMMON/A,B,P2,P1,D,NOT,NT2
C COMMON/B/STORE,IDMAT,ID,NBR
C COMMON/D/R1,R2

```



```

C 180 W( 1, NOT
DO 200 I = 1, NT2
  I1 = I
  J2 = MOD(I1, 4) + 1
  NO = 0
GO TO(189, 179, 185, 179), J2
C 185 IDENTIFY ELEMENTS OF ODD ROWS (ALWAYS 1, 3, 5, ...)
C
179 J = -1
180 J = J + 2
  IF(J.GT.MAX)GO TO 200
  NO = NO + 1
  IDMAT(I, NO) = J
GO TO 180
C 185 IDENTIFY ELEMENTS OF EVEN ROWS
C
185 IF(R.NE.1.0)GO TO 189
C 185 ELEMENTS ARE 4, 8, 12, ...
C
  J = -2
GO TO 190
189 J = 0
C 185 ELEMENTS ARE 2, 4, 6, ... OR 2, 6, 10, ...
C
190 J = J + NDE
  IF(J.GT.MAX)GO TO 200
  NO = NO + 1
  IDMAT(I, NO) = J
GO TO 190
200 NBR(1) = NO
250 CONTINUE
C
  IF(IDIAG.LT.2)RETURN
  WRITE(6, 300)
  MAX = 1H
  JZ = 11 - SC
  IF(NT2.LT.JZ)JZ = NT2
DO 290 I = 1, JZ
  NO = NBR(I)
290 WRITE(6, 301)NO, ((MAX, ID(I), IDMAT(I, J)), J=1, NO)
C
  IF(JZ.EQ.NT2)RETURN
  JZ = JZ + 1
DO 295 I = JZ, NT2
  NO = NBR(I)
295 WRITE(6, 302)NO, ((MAX, ID(I), IDMAT(I, J)), J=1, NO)
C 300 FORMAT(/, ' TABLE OF INDICES (L.S) FOR REQUIRED ',
1. R(L.S)')
301 FORMAT(16, 3X, 10(A1, '( ', I1, ', ', I1, ', ', 3X))
302 FORMAT(16, 2X, 10(A1, '( ', I2, ', ', I1, ', ', 3X))
C
  RETURN
END
SUBROUTINE CHEBY(NOP, SC)
C
  INTEGER S, SC, SMAX
  DIMENSION STORE(200, 42), IDMAT(20, 10), ID(20), NBR(20)
  DIMENSION R1(10, 10), R2(20, 10)
  DIMENSION BUST(11), BIST(21), BKST(21)
  COMMON/A/B, P2, P1, D, NOT, NT2

```

```

COMMON/B/STORE, IDMAT, ID, NBR
C ON/C/P,OMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,BI,BIM,L,S
C 3N/D/R1,R2
CHEBYSHEV INTEGRATION TO PRODUCE R1(L,S) AND R2(L,S) (SC = 1)
OR T1(L,S) AND T2(L,S) (SC = 2)

PTS = NOP
W = PI/(PTS*2.0)
P = SORT(P2)
OMP = SORT(1.0 - P2)
LMAX = ID(NT2) + 1
SMAX = ID(NT2) + 1
ORD = 0.0
NDR = 0

ZERO ALL THE REQUIRED MATRIX ELEMENTS IN R1 AND R2

DO 55 I = 1,NT2
NO = NBR(I)
IF(MD.EQ.0)GO TO 55
DO 50 J = 1,NO
JK = IDMAT(I,J) - SC + 2
IF(I.GT.NOT)GO TO 50
R1(I,JK) = 0.0
50 R2(I,JK) = 0.0
55 CONTINUE

LOOP DO 101 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
LOOP DO 101 I.... CORRESPONDS TO L
LOOP DO 100 J.... CORRESPONDS TO S

DO 101 JJJ = 1,NOP
RH = STORE(JJJ,1)
DR = STORE(JJJ,2)

CALCULATE AND STORE THE J BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT
JO IS STORED AS BJST(1), J1 IS STORED AS BJST(2), ETC.

ARG = PI*B*RH*SORT(1.0 - P2)
CALL BESJ(ARG,ORD,SMAX,BJST,NZ)

CALCULATE AND STORE THE I BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT.
IO IS STORED AS BIST(1), I1 IS STORED AS BIST(2), ETC.

ARG = PI*B*RH*SORT(P2)
CALL BESJ(ARG,ORD,1,LMAX,BIST,NZ)

CALCULATE AND STORE THE K BESSEL FUNCTIONS NEEDED IN SUBROUTINE INT.
KO IS STORED AS BKST(1), K1 IS STORED AS BKST(2), ETC.

CALL BESKN(ARG,NOR,1,LMAX,BKST,NZ)

CALCULATE R1(L,S) AND R2(L,S) OR T1(L,S) AND T2(L,S)

DO 101 I = 1,NT2
NO = NBR(I)
IF(MD.EQ.0)GO TO 101
L = ID(I)
LM = L - 1
IF(L.EQ.0)LM = 1
BK = BKST(L+1)
BKM = BKST(LM+1)
BI = BIST(L+1)
BIM = BIST(LM+1)

```

```

IF(L.GT.0)GO TO 70
C 1.0
S 0.0
GO TO 72
70 CL = STORE(JJJ,2*L+1)
SL = STORE(JJJ,2*L+2)
72 CONTINUE
C
DO 100 J = 1,NO
S = 10MAT(I,J)
JK = S - SC + 2
BJ = BJST(S+1)
BJM = -BJST(2)
IF(S.GT.0)GO TO 80
CS = 1.0
SS = 0.0
GO TO 82
80 CS = STORE(JJJ,2*S+1)
SS = STORE(JJJ,2*S+2)
BJM = BJST(S)
82 CONTINUE
C
CALL INT(SC,RA,RB)
R2(I,JK) = R2(I,JK) + RB*W
IF(I.GT.NOT)GO TO 100
R1(I,JK) = R1(I,JK) + RA*W
100 CONTINUE
101 CONTINUE
C
END
SUBROUTINE MATR(NOP,SC,ND)
C
INTEGER SC
DIMENSION R1(10,10),R2(20,10)
DIMENSION SYM(10,10),ASY(10,10),C(10)
DIMENSION ALPHA(10,20),BST(31)
COMMON/A/B,P2,P1,D,NOT,NT2
COMMON/O/R1,R2
COMMON/E/SYM,ASY,C
COMMON/F/ALPHA,BST
C
CALCULATE THE COUPLING COEFFICIENTS ALPHA (SC = 1) OR BETA (SC = 2);
STORE AS ALPHA(I,J)
ARG = PI*B*SQRT(P2)*D
C
CALCULATE THE K BESSEL FUNCTIONS OF ORDER 0 THROUGH MAX = NOT + NT2,
AND STORE THEM WITH INDICES 1 THROUGH MAX + 1
MAX = NOT + NT2 + 1
I = 0
CALL BESKN(ARG,I,1,MAX,BST,J)
C
IF(SC.GT.1)GO TO 150
C
THIS SECTION CALCULATES ALPHA(K,L)
WE NEED THE K-BESSEL FUNCTIONS OF ORDER K - L AND ORDER K + L. FOR THE
COSINE SOLUTION, 0 <= K,L <= (NOT-1). TAKING K = I - 1, L = J - 1,
WE GET K+L = I + J - 2, K - L = I - J. HOWEVER SINCE THESE START AT
ZERO, WE NEED MM = I + J - 1 AND NN = I - J + 1 AS RETRIEVAL INDICES
FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
DO 100 I = 1,NOT
EPS = 2.0
IF(1.EQ.1)EPS = 1.0

```



```

DO 100 J = 1,NT2
  I = J - 1
  IF(I.GE.J)NN = I - J + 1
  IF(J.GT.I)NN = J - I + 1
  ALPHA(I,J) = EPS*(BST(MM) + BST(MN))/PI
GO TO 200

C
C
C THIS SECTION CALCULATES BETA(K,L).
C WE NEED THE K-BESSEL FUNCTIONS OF ORDER K - L AND K + L. FOR THE SINE
C EXPANSION, 1 <= K,L <= NOT. TAKING K = I AND L = J SO THAT K + L
C = I + J AND K - L = I - J, WE NEED MM = I + J + 1 AND NN = I - J + 1
C AS RETRIEVAL INDICES FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
C
C
C NOTE THAT EVEN THOUGH WE CALCULATE BETA(K,L) (SINE SOLUTION), WE STORE
C THE RESULTS IN ARRAY ALPHA.
C
150 CONTINUE
DO 170 I = 1,NOT
  DO 170 J = 1,NT2
    MM = I + J + 1
    IF(I.GE.J)NN = I - J + 1
    IF(J.GT.I)NN = J - I + 1
    ALPHA(I,J) = 2.0*(BST(MM) - BST(MN))/PI
170 CONTINUE
IF(MD.GT.1)GO TO 280

C
C CALL SUBROUTINE CHEBY TO DO THE LINE INTEGRALS TO FIND R1 AND R2
C
C CALL CHEBY(NOP,SC)
C
C CALCULATE THE ELEMENTS OF THE MATRIX FROM THE ELEMENTS OF R1 AND
C SUMS OF ELEMENTS OF R2.
C
280 CONTINUE
DO 290 I = 1,NOT
  DO 290 J = 1,NOT
    SYM(I,J) = R1(I,J)
    290 ASY(I,J) = R1(I,J)
  C
  SIGN = 1.0
  IF(SC.EQ.2)SIGN = -1.0
  DO 300 I = 1,NOT
    DO 300 J = 1,NOT
      SUM = 0.0
      DO 295 MM = 1,NT2
        295 SUM = SUM + ALPHA(1,MM)*R2(MM,J)
      SYM(I,J) = SYM(I,J) + SUM*SIGN
      300 ASY(I,J) = ASY(I,J) - SUM*SIGN
    RETURN
  END
  SUBROUTINE INT(SC,R1,R2)

C
C EVALUATE THE INTEGRANDS OF THE LINE INTEGRALS WHICH BECOME THE
C MATRIX ELEMENTS R1(L,S) AND R2(L,S)
C
C
C INTEGER S,SC
COMMON/A/B,P2,PI,D,NOT,NT2
COMMON/C/P,OMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,BI,BIM,L,S
C
XS = S
XL = L
EPS = 1.0
IF(L.GT.0)EPS = 2.0
TERM = BK*BJM*OMP + P*BJ*BKM
TERM = 2.0*B*RH*TERM
TERM = BI*BJM*OMP - P*BJ*BIM

```

```

C      HV = PI*B*RH*HERM
C      I (.EQ.S) GO TO 90
C      DD = L - S
C      TERM = TERM + 2.0*DD*BK*BJ/PI
C      HERM = HERM + DD*BI*BJ
C      90 F = EPS*TERM
C      H = EPS*HERM
C
C      NOTE THAT I=-(L+1) IS NOT INCLUDED IN F-
C      NOTE THAT I=0L IS NOT INCLUDED IN H
C
C      E = 0.0
C      G = 0.0
C      IF(L.EQ.S) GO TO 100
C      E = 2.0*EPS*BK*BJ*DR/(RH*PI)
C      G = EPS*BI*BJ*DR/RH
C
C      NOTE THAT I=-(L+1) IS NOT INCLUDED IN E
C      NOTE THAT I=0L IS NOT INCLUDED IN G
C
C      100 CONTINUE
C
C      IF(SC.EQ.2) GO TO 110
C      TERM = XS*CLS - XL*SLS*CS
C      R1 = 4.0*(F*CLS*CS + E*TERM)
C      R2 = 4.0*(H*CLS*CS + G*TERM)
C      RETURN
C
C      110 TERM = XS*SLS*CS - XL*S*S*CL
C      R1 = 4.0*(F*SLS*SS - E*TERM)
C      R2 = 4.0*(H*SLS*SS - G*TERM)
C      RETURN
C      END
C      SUBROUTINE MATINV(A,N,B,DETER,M)
C
C      DIMENSION A(10,10),B(10,1),INDEX(10,3)
C      10 DETER = 1.0
C      15 DO 20 J = 1,N
C      20 INDEX(J,3) = 0
C      30 DO 550 I = 1,N
C
C      SEARCH FOR PIVOT ELEMENT
C
C      40 AMAX = 0.
C      45 DO 105 J = 1,N
C      IF (INDEX(J,3)-1) 60,105,60
C      60 DO 100 K = 1,N
C      IF (INDEX(K,3)-1) 80,100,740
C      80 IF(AMAX-ABS(A(J,K))) 85,100,100
C      85 IROW = J
C      90 ICOLUM = K
C      AMAX = ABS(A(J,K))
C      100 CONTINUE
C      105 CONTINUE
C      IF(AMAX.EQ.0.0)GO TO 750
C      INDEX(ICOLUM,3) = INDEX(ICOLUM,3) + 1
C      260 INDEX(I,1) = IROW
C      270 INDEX(I,2) = ICOLUM
C
C      INTERCHANGE ROWS TO PUT PIVOT ELEM.IN DIAG.
C
C      130 IF(IROW=ICOLUM) 140,310,140
C      140 DETER = -DETER
C      150 DO 200 L = 1,N
C      160 SWAP = A(IROW,L)
C      170 A(IROW,L) = A(ICOLUM,L)

```

```

200 A(I,COLUMN,L) = SWAP
   I( ) 310,310,210
210 DO 350 L = 1,M
220 SWAP = B(IRON,L)
230 B(IRON,L) = B(ICOLUMN,L)
250 B(ICOLUMN,L) = SWAP
C
C   DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
310 PIVOT = A(ICOLUMN,ICOLUMN)
   DETER = DETER-PIVOT
330 A(ICOLUMN,ICOLUMN) = 1.0
340 DO 350 L = 1,N
350 A(ICOLUMN,L) = A(ICOLUMN,L)/PIVOT
355 IF (M) 360,360,360
360 DO 370 L = 1,M
370 B(ICOLUMN,L) = B(ICOLUMN,L)/PIVOT
C
C   REDUCE NON PIVOTS ROWS
C
380 DO 550 LI = 1,N
390 IF (LI-ICOLUMN) 400,550,400
400 I = A(LI,ICOLUMN)
420 A(LI,ICOLUMN) = 0.
430 DO 450 L = 1,N
450 A(LI,L) = A(LI,L)-A(ICOLUMN,L)*I
455 IF (M) 550,550,460
460 DO 500 L = 1,M
500 B(LI,L) = B(LI,L)-B(ICOLUMN,L)*I
550 CONTINUE
C
C   INTERCHANGE COLUMNS
C
600 DO 710 I = 1,N
610 L = N + 1 - I
620 IF (INDEX(L,1)-INDEX(L,2)) 630,710,630
630 JROW = INDEX(L,1)
640 JCOLUMN = INDEX(L,2)
650 DO 705 K = 1,N
660 SWAP = A(K,JROW)
670 A(K,JROW) = A(K,JCOLUMN)
700 A(K,JCOLUMN) = SWAP
705 CONTINUE
710 RETURN
C
750 DETER = 0.0
   RETURN
   END
C
C   SUBROUTINE PERIM(N,R,PHI,RHO,DRDP)
C
C   EVALUATE RHO(PHI) AND D(RHO)/D(PHI) ON THE PERIMETER.
C
C   REAL N
C
C   IF (R.NE.1.0.OR.N.NE.1.0)GO TO 100
   RHO = 1.0
   DRDP = 0.0
   RETURN
C
C   100 CONTINUE
   CP = COS(PHI)
   SP = SIN(PHI)
   PN = 2.0*N

```

DD = (CP/R)**PN + SP**PN
 PN = 1.0/PN
 R = 1.0/(DD**PN)
 PN = 2.0*N - 2.0
 AA = SP**PN - ((CP/R)**PN)/(R*R)
 ORDP = -RHO*SP*CP*AA/DD
 RETURN
 END

C

Program Listing

- TWOGS-

(Reference: Vol.I,, Section II.3.3)


```

C NL2 VALUE OF 1/2 FROM PREVIOUS CASE (HELPS AVOID REPEATING CALCULATIONS)
C NM4 # OF NONZERO VALUES OF D IN DST (LE.6)
C NOT # OF INTEGRATION POINTS (READ IN)
C NP2 # OF TERMS IN EXPANSIONS (READ IN)
C NR # OF VALUES OF P2 TAKEN TO CONSTRUCT TABLE (READ IN)
C NT2 # OF ROOTS OF DET(RLS) LOCATED
C PHI HIGHEST ALLOWED VALUE OF INDEX L IN R2(L,S) AND R3(L,S)
C P1 ANGLE IN (RHO,PHI) COORDINATE SYSTEM
C P2 LARGEST VALUE OF P2 USED IN TABLE (READ IN)
C P1N SMALLEST VALUE OF P2 USED IN TABLE (READ IN)
C PR ARRAY FOR STORING ALPHANUMERIC PROGRAM DATA
C P2 PROPAGATION CONSTANT 1 DESIRED RESULT)
C R1 ASPECT RATIO. GUIDE # 1 (READ IN)
C R2 ASPECT RATIO. GUIDE # 2 (READ IN)
C RHO RADIUS IN (RHO,PHI) COORDINATE SYSTEM
C RHST ARRAY FOR STORING RHO AND DRDP (AS FNS OF PHI) FOR EACH PERIMETER
C RL1 VALUE OF R1 FROM PREVIOUS CASE (HELPS AVOID REPEATING CALCULATIONS)
C RL2 VALUE OF R2 FROM PREVIOUS CASE (HELPS AVOID REPEATING CALCULATIONS)
C RLS ARRAY OF COEFFICIENTS WHOSE DETERMINANT IS TO BE FOUND
C ROOM ARRAY FOR STORING INFORMATION USED IN PRINTING RESULTS AT THE END
C ROOT ARRAY FOR STORING OUTPUT INFORMATION, INCLUDING VALUES OF P2
C RS1 * ARRAY CONTAINING THE SUBMATRIX R1(L,S)
C RS2 * ARRAY CONTAINING THE SUBMATRIX R2(L,S)
C RS3 * ARRAY CONTAINING THE SUBMATRIX R3(L,S)
C RS4 * ARRAY CONTAINING THE SUBMATRIX R4(L,S)
C SC INDICATES SINE OR COSINE EXPANSION (INTEGER) (READ IN)
C STORE ARRAY FOR STORING FUNCTIONS USED REPEATEDLY AND DEPENDING ONLY ON PHI
C SZ RATIO OF Y-DIMENSION OF GUIDE # 2 TO THAT OF GUIDE # 1 (READ IN)
C SZL VALUE OF SZ FROM PREVIOUS CASE (HELPS AVOID REPEATING CALCULATIONS)
C T0 TABLE INITIAL TABLE OF VALUES OF ROOTS FOR DIFFERENT VALUES OF P2
C T1 INITIAL CP TIME
C T2 LATER CP TIME
C * IDENTIFIES VARIABLES NOT USED IN MAIN PROGRAM, ONLY PRINTED
C (IF IDIAG > 3)
C *****
C INITIALIZE CONSTANTS FOR ENTIRE JOB
C *****
C PI = 3.1415926535898
C 2 CALL DATE(PN)
C CALL TIME(W)
C RL1 = 0.0
C RL2 = 0.0
C NL1 = 0.0
C NL2 = 0.0
C SZL = 0.0
C LSC = 0
C NOT = # OF TERMS TAKEN BEFORE TRUNCATION
C NT2 = LARGEST VALUE TAKEN FOR L IN R2(L,S)
C ACCURACY OF HALF-INTERVAL SEARCH IS 10**(-1)
C NP2 = # OF VALUES OF P2 IN THE INITIAL TABLE
C NP2 = # OF INTEGRATION POINTS IN THE RANGE (0,PI/2)
C IDIAG = DIAGNOSIS CODE: 0 = NORMAL CONDITION (DEFAULT)
C 1 = INITIAL TABLE, FINAL MATRIX PRINTED
C 2 = SUMMARY PRINTED WITH MORE SIG FIGURES
C 3 = DETAILS OF ROOT-LOCATION PRINTED
C 4 = R1, R2, ALPHA, R2*ALPHA PRINTED
C 5 = KN(D) PRINTED AFTER ROOTS FOUND
C DEF: NOT
C DEF: 1.0E-3
C DEF: 10
C DEF: 50
C * IF THE CROSS-SECTION IS CIRCULAR, THE PHI-INTEGRATIONS CAN BE DONE IN
C * CLOSED FORM. TO SAVE TIME ONE CAN SET NOP = 1, LEAVING ONLY ONE "INTE-
C * GRATION POINT". IN THIS CASE, ADJUSTMENTS IN THE VALUES OF THE TRIG
C * FUNCTIONS ARE MADE AT STATEMENT 87 SO THAT THE INTEGRAL WILL COME OUT

```



```

( RIGHT. NOTE THAT IF NOP = 1. NOT IS MUST ALSO EQUAL 1 ( ELSE THE OFF-
  DIAGONAL MATRIX ELEMENTS MAY BE NONZERO AND THEREFORE . CORRECT.

RETURN TO STATEMENT 2 ONLY AFTER DUMPING RESULTS FOR ALL THESE CASES.
A BLANK CARD HERE TERMINATES PROGRAM EXECUTION.

```

```

READ(5,100)NOT,NT2,I,NP2,NOP,IDIAG
IF(NOT.EQ.0)STOP
WRITE(6,130)
WRITE(6,123)
WRITE(6,101)N,W
IF(NOT.GT.10)NOT = 10
NDM = 2*NOT
IF(NT2.GT.20)NT2 = 20
IF(NT2.LT.10)NT2 = NOT
IF(1.EQ.0)I = 3
IF(1.GT.12)I = 12
ACC = 10.0*(-1)
IF(NP2.LT.1)NP2 = 10
IF(NP2.GT.50)NP2 = 50
IF(NOP.GT.200)NOP = 200
IF(NOP.EQ.0)NOP = 50
IF(NOP.EQ.1)NT2 = NOT

```

```

PRINT THE FIXED INFORMATION READ FROM THE FIRST DATA CARD

```

```

WRITE(6,102)NOT
WRITE(6,129)NT2
WRITE(6,104)I
WRITE(6,108)NP2
WRITE(6,105)NOP
WRITE(6,103)IDIAG

```

```

SC = 1 IF COSINE TERMS ARE TO BE USED; 2 IF SINE TERMS ARE DESIRED
D = CENTER SEPARATION OF GUIDES, IN UNITS OF SEMIMINOR AXIS DEF: TOUCH
N1 = INDEX OF SUPERELLIPSE # 1 (1 = ELLIPSE, > 30 = SQUARE, ETC.) DEF: 1
R1 = ASPECT RATIO OF GUIDE # 1 DEF: 1
N2 = INDEX OF SUPERELLIPSE # 2 DEF: 1
R2 = ASPECT RATIO OF GUIDE # 2 DEF: 1
S2 = RATIO OF SIZE OF GUIDE # 2 TO THAT OF GUIDE # 1 DEF: 1
B = ASSUMED VALUE OF THE BINDING CONSTANT, B DEF: 1
PMA, PMIN = RANGE OF VALUES OF P2 TO BE SEARCHED DEF: 1, 0

```

```

ONE DATA CARD CORRESPONDS TO ONE INDIVIDUAL CASE

```

```

RETURN TO STATEMENT 1 IMMEDIATELY AFTER EACH CASE IS COMPLETE. A BLANK
CARD READ AT STATEMENT 1 CAUSES RESULTS FOR ALL CASES TO BE DUMPED
(PRINTED) AND CONTROL WILL THEN GO TO STATEMENT 2. A BLANK CARD THERE
TERMINATES THE JOB EXECUTION, SO TWO BLANK CARDS SHOULD FOLLOW THE
LAST DATA CARD. IF A DATA CARD CONTAINS MORE THAN ONE NONZERO VALUE OF
DST (THE GUIDE SPACINGS, D) THE PROGRAM WILL RETURN TO STATEMENT 6 TO
PROCESS THOSE PAST THE FIRST. THIS IS A MORE EFFICIENT PROCEDURE THAN
READING EACH VALUE OF D SEPARATELY FROM OTHERWISE IDENTICAL DATA CARDS.

```

```

JDEX = 0
1 READ(5,121)SC,DST(1),N1,R1,N2,R2,S2,B,PMA,PMIN,(DST(I),I=2,6)
IF(R1.LE.0)GO TO 999

```

```

CALL SECOND(TO)
IF(PMA.EQ.0.0)PMA = 1.0
IF(R2.LE.0.0)R2 = 1.0
IF(N1.LE.0.0)N1 = 1.0
IF(N2.LE.0.0)N2 = 1.0

```

```

IF(R.LE.0.0)B = 1.0
  SC = 1
  IF(CGT.2)SC = 2
  IF(SZ.LE.0.0)SZ = 1.0
.....
COUNT THE NUMBER, NMAX, OF CASES (DIFFERENT D) ON THE LAST DATA CARD
ND = 0
D = R1 + R2*SZ
DO 5 I = 1,6
  IF(DST(I).EQ.0.0)GO TO 6
  IF(DST(I).LT.D)DST(I) = D
  NMAX = I
5 CONTINUE
6 ND = ND + 1
D = DST(ND)
IF(JDEX.GE.20)GO TO 999
JDEX = JDEX + 1
.....
PRINT INPUT DATA
WRITE(6,111)
WRITE(6,123)
WRITE(6,111)
WRITE(6,112).DEX,D.M1,R1,N2,R2,SZ,B
IF(PMAX.NE.1.0.OR.PMIN.NE.0.0)WRITE(6,110)PMAX,PMIN
.....
INITIALIZE STORAGE ARRAYS FOR THIS CASE
DO 70 I = 7,14
  DO 69 K = 1,NDM
    CFF(JDEX,I-6,K) = 0.0
70 ROOT(JDEX,I) = 0.0
    ROOT(JDEX,3) = N1
    ROOT(JDEX,4) = R1
    ROOM(JDEX,2) = SZ
    ROOM(JDEX,3) = N2
    ROOM(JDEX,4) = R2
    ROOT(JDEX,5) = D
    ROOT(JDEX,6) = B
    IROOT(JDEX,1) = 0
    IROOT(JDEX,2) = SC - 1
    IROOT(JDEX,3) = NOT + SC - 2
.....
IDENTIFY CROSS-SECTIONS FOR THIS CASE
NG = 1
R = R1
N = N1
J = SC + 5
89 I = 4
  IF(N.GT.1.0)GO TO 90
  IF(N-.5)94,95,96
94 I = 9
  GO TO 92
95 I = 8
  GO TO 92
96 IF(N.NE.1.0)GO TO 92

```

```

      I( 2 ..EQ.1.0)I = 1
90 IF(N.LT.30)GO TO 92
      I = 5
      IF(R.EQ.1.0)I = 3
92 WRITE(6,109)NG,PR(I)
      IF(NG.EQ.2)GO TO 98
      ROOT(JDEX,2) = PR(J)
      ROOT(JDEX,1) = PR(I)
      IF(NOP.EQ.1.AND.I.NE.1)WRITE(6,131)
      NG = 2
      R = R2
      N = N2
      GO TO 89
98 WRITE(6,133)PR(J)
      ROOM(JDEX,1) = PR(I)
      IF(NOP.EQ.1.AND.I.NE.1)WRITE(6,131)
      .....
      CALCULATE AND STORE COS(M*PHI) AND SIN(M*PHI) AS STORE(I,2*M-1)
      AND STORE(I,2*M), RESPECTIVELY.
      IF(JDEX.GT.1)GO TO 76
      PN = NOP
      W = PI/(PN*2.0)
      DO 74 I = 1,NOP
      PN = 2*(NOP - I) + 1
      PHI = W*PN/2.0
      DO 74 M = 1,NT2
      PN = M
      PN = PN*PHI
      STORE(I,2*M-1) = COS(PN)
      STORE(I,2*M) = SIN(PN)
74 IF(NOP.GT.1)GO TO 76
      IF NOP = 1, ASSUME TWO CIRCLES, SET TRIG FUNCTIONS EQUAL TO CONSTANTS
      CHOSEN SO THAT THE ANALYTICAL RESULTS FOR THE INTEGRALS ARE OBTAINED.
      DO 75 M = 1,NT2
      STORE(I,2*M-1) = SORT(.5)
      STORE(I,2*M) = SORT(.5)
75 DETERMINE THE INDEX PAIRS (L,S) OF ALL THE MATRIX ELEMENTS R(L,S) NEEDED
      BY CALLING SUBROUTINE REL
      76 NG = 1
      I = LSC
      CALL CODE(N1,R1,NL1,RL1,SC,LSC,LST)
      LSC = I
      IF(LST.EQ.1.OR.LST.EQ.3)GO TO 78
      CALL REL(N1,R1,SC,NG,IDIAG)
      NG = 2
      CALL CODE(N2,R2,NL2,RL2,SC,LSC,J)
      IF(J.EQ.1.OR.J.EQ.3)GO TO 80
      CALL REL(N2,R2,SC,NG,IDIAG)
80 IF(SZ.EQ.SZL)GO TO 81
      SZL = SZ
      IF(J.GE.3)GO TO 81
      J = J + 2
      81 CONTINUE
      IF(LST.LT.3.AND.J.LT.3)GO TO 87
      PN = NOP
      W = PI/(PN*2.0)

```

```

C          CALCULATE AND STORE RHO1(PH1) AS RHST(1,1), I = 1,NOP
C          CALCULATE AND STORE RHO2(PH1) AS RHST(1,3), I = 1,NOP
C          CALCULATE AND STORE DIRHO1(D,PH1) AS RHST(1,2), I = 1,NOP
C          CALCULATE AND STORE DIRHO2(D,PH1) AS RHST(1,4), I = 1,NOP
C
C      DO 85 I = 1,NOP
C      PN = 2*(NOP - I) + 1
C      PHI = W*PN/2.0
C      CALL PERIN(N1,R1,PH1,RHO,DRDP)
C      RHST(1,1) = RHO
C      RHST(1,2) = DRDP
C      CALL PERIN(N2,R2,PH1,RHO,DRDP)
C      RHST(1,3) = RHO*SZ
C      85 RHST(1,4) = DRDP*SZ
C
C      .....
C      CONSTRUCT TABLE OF VALUES OF DET(RLS) VS P2, GIVEN B. TO SAVE TIME,
C      (SINCE RS1(L,S) - RS4(L,S) ARE INDEPENDENT OF D) THIS
C      IS DONE FOR EACH OF THE VALUES OF D READ INTO ARRAY DST WHEN THE
C      COUNTER, ND, IS EQUAL TO 1. THIS SECTION IS SKIPPED WHEN ND.GT.1.
C
C      87 LST = NP2 + 1
C      IF(ND.GT.1)GO TO 205
C      IF(IDIAG.GT.0)WRITE(6,106)
C      PN = NP2
C      W = (PMAX - PMIN)/PN
C      P2 = PMAX
C      IF(PMAX.EQ.1.0)P2 = 1.0 - W
C      IF(NMAX.GT.1.AND.IDIAG.GT.0)WRITE(6,132)(DST(I),I=1,NMAX)
C      NR = NMAX + 1
C      DO 200 M = 1,LST
C      TABLE(M,1) = P2
C      DO 190 I = 1,NMAX
C      D = DST(I)
C      CALL MATR(NOP,SC,I)
C      CALL MATINV(RLS,NOM,C,DET,0)
C      190 TABLE(M,I+1) = DET
C      IF(IDIAG.GT.0)WRITE(6,107)M,(TABLE(M,I),I=1,NR)
C      P2 = PMAX - W*PN
C      200 IF(P2.LT.W)P2 = W
C      IF(IDIAG.GT.0)WRITE(6,114)
C      D = DST(I)
C
C      CALL SECOND(T1)
C      T1 = T1 - TO
C      WRITE(6,115)T1
C      GO TO 209
C
C      WHEN ND.GT.1, RECOVER THE TABLE OF VALUES OF DET(RLS)
C      VS P2 CALCULATED EARLIER.
C
C      205 DO 206 M = 1,LST
C      206 TABLE(M,2) = TABLE(M,ND+1)
C
C      .....
C      MAKE TENTATIVE IDENTIFICATION OF ROOTS OF EQUATION DET(RLS) = 0
C
C      209 NR = 0
C      IF(TABLE(1,2).NE.0.0)GO TO 210
C      NR = 1
C      INDEX(NR) = 1001
C      210 DO 220 M = 2,LST
C      PN = ABS(TABLE(M,2))

```



```

LST = 0
258 MAX = 1,NT2
259 ALPHA(J,MAX)*RS2(MAX,K)*(-1.0)**(MAX*SC)
IF(DN.EQ.0.0)GO TO 258
LST = LST + 1
CFF(LST,8,20) = DN
258 CONTINUE
259 WRITE(6,118)XXX,(CFF(MAX,8,20),MAX=1,LST)
260 WRITE(6,114)
XXX = 3HRA3
DO 263 J = 1,NOT
DET = (-1.0)**(J*SC-1)
DO 262 K = 1,NOT
LST = 0
DO 261 MAX = 1,NT2
DN = ALPHA(J,MAX)*RS3(MAX,K)*DET
IF(DN.EQ.0.0)GO TO 261
LST = LST + 1
CFF(LST,8,20) = DN
261 CONTINUE
262 WRITE(6,118)XXX,(CFF(MAX,8,20),MAX=1,LST)
263 WRITE(6,114)
IF(IDIAG.LT.5)GO TO 265
XXX = 3HKN
J = NOT + NT2 + 1
WRITE(6,118)XXX,(BST(K),K=1,J)
WRITE(6,114)
C .....
C PRINT THE FINAL MATRIX, IF DESIRED: STORE ROOT: CALCULATE A FINAL VALUE
C FOR THE DETERMINANT. PRINT THE COEFFICIENTS, AND ALSO CONVERT THE
C COEFFICIENTS FOR THE FIELD EXPANSION IN THE 2ND GUIDE TO THE
C PROPER FORM: THAT IS, CHANGE THE SIGN OF THE ODD COEFFICIENTS.
C .....
265 IF(1.LE.8)ROOT(JDEX,I*6) = P2
IF(IDIAG.LT.1)GO TO 253
XXX = 3HRLS
WRITE(6,114)
DO 252 J = 1,NDM
252 WRITE(6,118)XXX,(RLS(J,K),K=1,NDM)
253 WRITE(6,114)
CALL MATINV(RLS,NDM,C,DET,0)
WRITE(6,125)XXX,JDEX,I,P2,PN,ID,DET
J = NOT + 3 - SC
DO 248 K = J,NDM,2
248 C(K) = -C(K)
WRITE(6,122)(C(K),K=1,NDM)
WRITE(6,114)
IF(DET.EQ.0.0)WRITE(6,113)
IF(1.GT.8)GO TO 250
DO 249 J = 1,NDM
249 CFF(JDEX,I,J) = C(J)
250 CONTINUE
C .....
C CALL SECOND(T1)
C T1 = T1 - TO
C WRITE(6,126)T1
C IF(ND-NMAX)6,1,1
C .....
C RETURN TO EITHER PROCESS ANOTHER VALUE OF D FROM THE LAST DATA CARD,
C OR TO READ A NEW CARD.
C .....

```

```
C      .[ .....X.....  
C  
C  
C    AFTER ALL CASES (20 OR LESS) HAVE BEEN COMPLETED, PRINT A TABLE  
C OF RESULTS GIVING THE ROOTS AND THE COEFFICIENT RATIOS FOR EACH CASE.  
C  
C 999 WRITE(6,123)  
C     WRITE(6,123)  
C     WRITE(6,123)  
C     DO 900 I = 1,JDEX  
C       MAX = IROOT(I,1) + 6  
C       IF(IDIAG.LE.1)WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),  
C         (IROOT(I,J),J=2,MAX)  
C       IF(IDIAG.GE.2)WRITE(6,127)I,ROOT(I,1),(IROOT(I,M),M=2,3),  
C         (IROOT(I,J),J=2,MAX)  
C 900 WRITE(6,134)(ROOM(I,M),M=1,4)  
C  
C   IF(IDIAG.EQ.0)GO TO 2  
C     WRITE(6,111)  
C     WRITE(6,119)  
C     DO 910 I = 1,JDEX  
C       MAX = IROOT(I,1)  
C       IF(MAX.EQ.0)GO TO 905  
C       WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6),  
C         (ICFF(I,K,1),K=1,MAX)  
C       WRITE(6,134)(ROOM(I,M),M=1,4),(CFF(I,J,2),J=1,MAX)  
C       LST = NDM - 2  
C       IF(LST.LE.2)GO TO 910  
C       DO 902 K = 3,NDM  
C       GO TO 910  
C 902 WRITE(6,120)(CFF(I,M,K),M=1,MAX)  
C  
C 905 WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),(ROOT(I,J),J=2,6)  
C     WRITE(6,134)(ROOM(I,M),M=1,4)  
C 910 CONTINUE  
C  
C   GO TO 2  
C  
C END  
C SUBROUTINE CODE(N,R,NL,RL,SC,LSC,LST)  
C  
C REAL N,NL  
C INTEGER SC  
C  
C FIND LST: CODE FOR DECIDING WHEN TO RECALCULATE INDICES (2),  
C PERIMETER (3), OR BOTH (4).  
C  
C   LST = 1  
C   IF(R.EQ.RL)/GO TO 71  
C  
C   FIND LST FOR CASES WITH R.NE.RL  
C  
C   LST = 3  
C   IF(R.NE.1.0.AND.RL.NE.1.0)GO TO 73  
C   GO TO 72  
C  
C 71 IF(N.EQ.NL)GO TO 73  
C  
C   FIND LST FOR CASES WITH R.EQ.RL  
C  
C   LST = 3  
C   IF(R.NE.1.0)GO TO 73  
C   IF(N.NE.1.0.AND.NL.NE.1.0)GO TO 73  
C 72 LST = 4  
C   GO TO 74  
C  
C 73 IF(SC.EQ.LSC)GO TO 74
```

```

C      Q      GE LST WHEN SC.NE.LSC
C      LST = LST + 1
74 LSC = SC
   RL = R
   NL = N
   RETURN
END
SUBROUTINE RTTY(TABLE,ACC,PN,NOP,SC,M,IO,IDIAG)
C
C      SUBROUTINE RTTY PRECISELY LOCATES THE ROOT SPECIFIED BY M (USING
C      ELEMENTS M AND M+1 IN THE ARRAYS TABLE(M,1) AND TABLE(M,2), AND THEN
C      DETERMINES THE FIELD EXPANSION COEFFICIENTS C(J) CORRESPONDING TO
C      THAT ROOT. ACC IS THE DESIRED ACCURACY OF THE ROOT, AND PN IS THE
C      DIFFERENCE IN THE ACTUAL BOUNDS OBTAINED. IO IS THE NUMBER OF FUN-
C      CTION EVALUATIONS NECESSARY. SC, IDIAG, AND NOP ARE AS IN MAIN. RLS, THE
C      FINAL MATRIX AT THE ROOT P2, IS ALSO RETURNED. (C, RLS, AND P2
C      ARE ALL IN COMMON BLOCKS.)
C
C      INTEGER SC
C      DIMENSION RLS(20,20),C(20)
C      DIMENSION TABLE(51,7),SINV(20,20)
C      COMMON/A,B,P2,P1,D,NOT,NT2
C      COMMON/E/RLS,C
C
C      700 FORMAT(/,' SEARCH IO=7X,'X0=,11X,'X1=,11X,'X2=,10X,'F0=,9X,'F1=,
C      19X,'F2=,11X,'X=,10X,'F=,10X,'DX=')
C      701 FORMAT(/,' $$$$ ERROR IN HINTS OR MULLER, IER =,13,' $$$$$')
C      IF(IDIAG.GE.3)WRITE(6,700)
C      K = 0
C      ID = 0
C      IF(M.LT.1000)GO TO 230
C      P2 = TABLE(M-1000,1)
C      CALL MATR(NOP,SC,1)
C      ID = 1
C      GO TO 248
C
C      230 G1 = TABLE(M+1,2)
C      G2 = TABLE(M,2)
C      X1 = TABLE(M+1,1)
C      X2 = TABLE(M,1)
C      DN = X2 - X1
C
C      IF G1 AND G2 ARE TWO OR MORE ORDERS OF MAGNITUDE DIFFERENT, USE
C      A HALF-INTERVAL SEARCH TO NARROW THE RANGE X1 TO X2
C
C      235 NW = -G1/G2
C      IF(NW.LT.100..AND.NW.GT..01)GO TO 240
C      IF(DN.LT.ACC)GO TO 240
C      DN = DN/3.0
C
C      CALL HINTS(X1,X2,G1,G2,DN,PN,NOP,SC,IO,IDIAG,IER)
C      IF(IER.GT.0)WRITE(6,701)IER
C
C      K = K + ID
C      IF(PN.EQ.0.0)GO TO 248
C      GO TO 235
C
C      MULLER'S METHOD TO GET THE ROOT PRECISELY
C
C      240 CALL MULLER(X1,X2,G1,G2,ACC,PN,NOP,SC,IO,IDIAG,IER)
C      IF(IER.GT.0)WRITE(6,701)IER
C
C      248 IO = IO + K

```



```

C 40 X = (XL + XU)/2.0
P = X
CALL MATR(NOP,SC,1)
CALL MATINV(RLS,NDM,C,F,0)
ID = ID + 1
IF(F*SIGN)50,200,60
50 XU = X
60 XL = X
GO TO 100
C
C EVALUATE THE NEW X FROM THE OLD X0, X1, AND X2
C
100 X1 = X
F1 = F
DD01 = (F1 - F0)/(X1 - X0)
DD12 = (F2 - F1)/(X2 - X1)
DD012 = (DD12 - DD01)/(X2 - X0)
W = DD01 + (X1 - X0)*DD012
U = DD012/W
GL = F1/W
U = 1.0 - 4.0*U*GL
IF(U.LT.0.0)GO TO 260
U = 2.0*GL/(1.0 + SORT(U))
X = X1 - U
IF(X.GT.X0.AND.X.LT.X2)GO TO 145
U = 1.0 - 4.0*GL*DD012/W
U = 2.0*GL/(1.0 - SORT(U))
X = X1 - U
145 CONTINUE
IF(X.GT.1.0.OR.X.LT.0.0)GO TO 270
P2 = X
CALL MATR(NOP,SC,1)
ID = ID + 1
PN = ABS(XU - XL)
IF(PN.LT.ACC)GO TO 210
IF(ID.GT.LST)GO TO 280
CALL MATINV(RLS,NDM,C,F,0)
C IF(IDIAG.GE.3)WRITE(8,700)ID,X0,X1,X2,F0,F1,F2,X,F,PN
C
C RESET THE UPPER OR LOWER BOUND ON THE ROOT
C
C IF(F*SIGN)150,200,155
150 XU = X
GO TO 160
155 XL = X
C
C CHANGE X0 OR X2 TO THE PREVIOUS X1; THE NEW X1 IS X
C
160 IF(F1*SIGN)165,200,170
165 X2 = X1
F2 = F1
GO TO 175
170 X0 = X1
F0 = F1
175 GL = ABS(U)
IF((F*F1).LT.0.0)GO TO 100
IF(GL.GT.ACC)GO TO 100
C
C IF POINTS ACCUMULATE CLOSELY ON ONE SIDE OF THE TRUE ROOT, A SINGLE
C SMALL JUMP MAY SPAN THE ROOT. THEREFORE CHANGE X BY A LITTLE LESS
C THAN ACC AND EVALUATE THE DETERMINANT. IF THIS FAILS (THAT IS, IF THE
C FUNCTION HAS THE SAME SIGN AGAIN) SPLIT THE INTERVAL IN HALF AND
C CONTINUE.

```

```

C
C
C      LINEAR EXTRAPOLATION PAST THE ANTICIPATED ZERO
C
      W = 1.0E-12
      IF(X.NE.X1)GO TO 180
      XT = W
      IF((F*SIGN).GT.0.0)XT = -W
      GO TO 185
180  XT = 2.0*F*(X1 - X1)/(F1 - F)
      GL = ABS(XT)
      IF(GL.GT.ACC)GO TO 100
      IF(GL.GT.W)GO TO 185
      XT = W*XT/GL
185  XT = X - XT
      P2 = XT
      CALL MATR(NOP,SC,1)
      CALL MATINV(RLS,NOM,C,FT,0)
      ID = ID + 1
      IF(IDIAG.GE.3)WRITE(6,701)ID,XT,FT
190  IF(FT.F)190,200,40
192  XU = XT
      X2 = XT
      F2 = FT
      IF(X-X1)100,205,100
193  XL = XT
      X0 = XT
      F0 = FT
      IF(X-X1)100,205,100
C
C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH MAY HAVE BEEN
C      ALTERED IN MATINV) IS RECOMPUTED.
C
200  PN = 0.0
      GO TO 209
C
C      IF X = X1, TAKE THIS AS THE ROOT
C
205  P2 = X
      PN = ABS(XU - XL)
C
C
209  CALL MATR(NOP,SC,1)
      ID = ID + 1
      NORMAL RETURN: PN,LT,ACC
C
210  IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X
      RETURN
C
C      ERROR RETURN 1: COMPLEX ROOT
C
260  IER = 1
      GO TO 299
C
C      ERROR RETURN 2: PROCEDURE IS OSCILLATING
C
270  IER = 2
      GO TO 299
C
C      ERROR RETURN 3: MORE ITERATIONS THAN HALF-INTERVAL SEARCH WOULD REQUIRE
C
280  IER = 3
      GO TO 210
C

```

```

C      ERROR RETURN 4: X0 > X2 INITIALLY
C      290 ILL = 4
C      299 ID = ID + 1
C      CALL MATR(NOP,SC,1)
C      RETURN
C      END
C      SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C      C
C      C      SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE THE ROOTS OF
C      C      THE EQUATION DET(RLS) = 0 OR DET(ITS) = 0 TO WITHIN A DESIRED ACCURACY
C      C      ACC. ID IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
C      C
C      C      INTEGER SC
C      C      DIMENSION RLS(20,20),C(20)
C      C      COMMON/A,B,P2,P1,D,NOT,NT2
C      C      COMMON/E/RLS,C
C      C
C      C      700 FORMAT(= HINTS:0,13,13X,2F13.10,11X,2E11.4,F13.10,2E11.4)
C      C
C      C      ID = 0
C      C      IER = 0
C      C      NDM = 2*NOT
C      C      100 PN = ABS(X2 - X1)
C      C
C      C      IF(PN.LT.ACC)RETURN
C      C      ID = ID + 1
C      C      IF(ID.GT.25)GO TO 230
C      C      P2 = (X1 + X2)/2.0
C      C      CALL MATR(NOP,SC,1)
C      C      CALL MATINV(RLS,NDM,C,G,0)
C      C
C      C      IF(IDIAG.GE.3)WRITE(6,700)ID,X1,X2,G1,G2,P2,G,PN
C      C
C      C      IF(G*(G1/ABS(G1)))120,210,130
C      C      120 X2 = P2
C      C      G2 = G
C      C      GO TO 100
C      C      130 X1 = P2
C      C      G1 = G
C      C      GO TO 100
C      C
C      C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE BEEN
C      C      ALTERED IN MATINV) IS RECOMPUTED.
C      C
C      C      210 PN = 0.0
C      C      CALL MATR(NOP,SC,1)
C      C      ID = ID + 1
C      C      RETURN
C      C
C      C      ERROR RETURN: TOO MANY ITERATIONS
C      C
C      C      230 IER = 6
C      C      RETURN
C      C      END
C      C      SUBROUTINE LOAD(SS,SINV,C,NOT,ID)
C      C
C      C      SUBROUTINE LOAD LOADS SS (NEE RLS) INTO ARRAY SINV, ELIMINATING
C      C      THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
C      C      IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
C      C      LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
C      C      INATED, THUS, SINV IS (NOT-1) BY (NOT-1) IN DIMENSION, AND C IS A VEC-
C      C      TOR (NOT-1) IN LENGTH.
C      C
C      C      REMEMBER, NOT IS WHAT IS CALLED NDM IN TWOGS. ALSO, ONLY THE FIRST
C      C      FOUR ELEMENTS OF EACH HALF OF THE DIAGONAL ARE SEARCHED FOR THE

```

```

C C
C C      SMALLEST VALUE.
C C      DIMENSION SS(20,20),SINV(20,20),C(20)
C C      G = 1.0E+321
C C      ID = 1
C C      K = NOT
C C      IF(K.GT.4)K = 4
C C      DO 200 J = 1,K
C C      M = ABS(SS(J,J))
C C      IF(M.GT.G)GO TO 200
C C      G = M
C C      ID = J
C C      200 CONTINUE
C C
C C      JJ = NOT/2 + 1
C C      K = JJ + 3
C C      IF(K.GT.NOT)K = NOT
C C      DO 210 J = JJ,K
C C      M = ABS(SS(J,J))
C C      IF(M.GT.G)GO TO 210
C C      G = M
C C      ID = J
C C      210 CONTINUE
C C
C C      JJ = 0
C C      DO 300 J = 1,NOT
C C      IF(J.EQ.ID)GO TO 300
C C      JJ = JJ + 1
C C      C(JJ) = -SS(J,ID)
C C      KK = 0
C C      DO 299 K = 1,NOT
C C      IF(K.EQ.ID)GO TO 299
C C      KK = KK + 1
C C      SINV(JJ,KK) = SS(J,K)
C C      299 CONTINUE
C C      300 CONTINUE
C C      RETURN
C C      END
C C      SUBROUTINE REL(N,R,SC,NG,IDIAG)
C C
C C      SUBROUTINE REL IDENTIFIES THOSE MATRIX ELEMENTS R(L,S) (COSINE EX-
C C      PANSION, SC = 1) OR T(L,S) (SINE EXPANSION, SC = 2) WHICH ARE NONZERO.
C C      THE VALUES OF THE ROW INDICES, L, ARE STORED IN VECTOR ID(1), AND
C C      THE VALUES OF THE INDICES S, NBR(1,NG) IN NUMBER, ARE THE ROWS OF
C C      MATRIX IDMAT(1,J,NG). NG REFERS TO GUIDE # 1 OR GUIDE # 2.
C C
C C      AS AN EXAMPLE, CONSIDER A FIVE-TERM COSINE EXPANSION.
C C      WITH R = 1, THE NONZERO ELEMENTS ARE R(0,0), R(0,4)
C C      R(1,1), R(1,3)
C C      R(2,2)
C C      R(3,1), R(3,3)
C C      R(4,0), R(4,4)
C C      NBR(1) = 2
C C      AND ID(1) = 0 IDMAT(1,J) = 0, 4
C C      1 3
C C      2 2
C C      3 1, 3
C C      4 0, 4
C C
C C      INTEGER SC
C C      REAL N
C C      DIMENSION STORE(200,40),RHST(200,4),IDMAT(20,10,2),ID(20)
C C      DIMENSION NBR(20,2)
C C      DIMENSION R1(10,10),R2(20,10),R3(20,10),R4(10,10)
C C      COMMON/A,B,P2,P1,D,NOT,M2
C C      COMMON/B/STORE,RHST,IDMAT,ID,NBR
C C      COMMON/C/R1,R2,R3,R4

```

```

C      { 10.EQ.2)GO TO 56
C
DO 50 I = 1,NT2
  DO 50 J = 1,NOT
    R3(I,J) = 0.0
  50 R2(I,J) = 0.0
  DO 55 I = 1,NOT
    DO 55 J = 1,NOT
      R4(I,J) = 0.0
  55 R1(I,J) = 0.0
C
DO 60 I = 1,NT2
  I1 = I + SC - 2
  60 ID(I) = I1
C
C      FIND INDICES FOR A CIRCULAR GUIDE
C
56 IF(N.NE.1.0.OR.R.NE.1.0)GO TO 70
DO 62 I = 1,NT2
  NBR(I,NG) = 1
  62 IDMAT(I,NG) = I + SC - 2
  IF(NT2.EQ.NOT)GO TO 250
  I1 = NOT + 1
  DO 68 I = 11,NT2
    68 NBR(I,NG) = 0
  GO TO 250
C
70 NDE = 2
  IF(R.EQ.1.0)NDE = 4
  IF(SC.GT.1)GO TO 150
C
C      COSINE EXPANSIONS; HIGHEST INDEX USED = MAX = NOT - 1
C
MAX = NOT - 1
DO 100 I = 1,NT2
  I1 = I - 1
  J2 = MOD(I1,4) + 1
  ND = 0
  GO TO (85,79,89,79),J2
C
C      IDENTIFY ELEMENTS OF ODD ROWS (ALWAYS 1,3,5,...)
C
79 J = -1
80 J = J + 2
  IF(J.GT.MAX)GO TO 100
  ND = ND + 1
  IDMAT(I,NG,NG) = J
  GO TO 80
C
C      IDENTIFY ELEMENTS OF EVEN ROWS
C
85 IF(R.NE.1.0)GO TO 89
  ELEMENTS ARE 0,4,8,...
  J = -4
  GO TO 90
C
C      ELEMENTS ARE 0,2,4,... OR 2,6,10...
C
89 J = -2
90 J = J + NDE
  IF(J.GT.MAX)GO TO 100
  ND = ND + 1

```



```

C      I( GER 5, SC, SMAX,
C      DIMENSION STORE(200,40), RHST(200,4), IDMAT(20,10,2), ID(20)
C      DIMENSION NBR(20,2)
C      DIMENSION R1(10,10), R2(20,10), R3(20,10), R4(10,10)
C      DIMENSION BJST(11), B1ST(21), BMST(21)
C      COMMON/A/B,P2,P1,D,NOT,NT2
C      COMMON/B/STORE,RHST,IDMAT,ID,NBR
C      COMMON/C/P,OMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,B1,B1M,L,S
C      COMMON/D/R1,R2,R3,R4
C
C      CHEBYSCHEV INTEGRATION TO PRODUCE R1(L,S) THROUGH R4(L,S) (SC = 1)
C      OR T1(L,S) THROUGH T4(L,S) (SC = 2)
C
C      PTS = NOP
C      W = PI/(PTS+2.0)
C      P = SORT(P2)
C      LMAX = SORT(1.0 - P2)
C      SMAX = ID(INT2) + 1
C      ORD = 0.0
C      NOR = 0
C
C      ZERO ALL THE REQUIRED MATRIX ELEMENTS IN R1, R2, R3, AND R4
C
C      DO 60 I = 1,NT2
C      NO = NBR(I,1)
C      IF(NO.EQ.0)GO TO 53
C      DO 50 J = 1,NO
C      JK = IDMAT(I,J,1) - SC + 2
C      IF(I.GT.NOT)GO TO 50
C      R1(I,JK) = 0.0
C      R3(I,JK) = 0.0
C      50 NO = NBR(I,2)
C      IF(NO.EQ.0)GO TO 60
C      DO 55 J = 1,NO
C      JK = IDMAT(I,J,2) - SC + 2
C      IF(I.GT.NOT)GO TO 55
C      R4(I,JK) = 0.0
C      55 R2(I,JK) = 0.0
C      60 CONTINUE
C
C      THE INTEGRATION LOOP DO 101 JJJ = 1,NOP IS ENTERED TWICE, ONCE
C      FOR THE FIRST GUIDE (NG = 1) AND ONCE FOR THE SECOND (NG = 2). THE
C      LOOP PRODUCES TWO COMPLETE ARRAYS EACH TIME: R1(L,S) AND R2(L,S)
C      OR R3(L,S) AND R4(L,S).
C
C      NG = 0
C      65 NG = NG + 1
C      IF(NG.GT.2)RETURN
C
C      LOOP DO 101 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
C      LOOP DO 101 I.... CORRESPONDS TO L
C      LOOP DO 100 J.... CORRESPONDS TO S
C
C      DO 101 JJJ = 1,NOP
C      RH = RHST(JJJ,2*NG-1)
C      DR = RHST(JJJ,2*NG)
C
C      CALCULATE AND STORE THE J BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT
C      JO IS STORED AS BJST(1), J1 IS STORED AS BJST(2), ETC.
C      ARG = PI*B*RH*OMP
C      CALL BESJ(ARG,ORD,SMAX,BJST,NZ)

```



```

C      CALCULATE AND STORE THE I BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT.
C      ;( 3 STORED AS BIST(1), 11 IS STORED AS BIST(2), ETC.
C      ARG = PI*8*RH*P
C      CALL BESJ(ARG,ORD,1,LMAX,BIST,NZ)
C
C      CALCULATE AND STORE THE K BESSEL FUNCTIONS NEEDED IN SUBROUTINE INT.
C      K0 IS STORED AS BKST(1), K1 IS STORED AS BKST(2), ETC.
C      CALL BESKN(ARG,NOR,1,LMAX,BKST,NZ)
C
C      CALCULATE R1(L,S) AND R2(L,S) (NG = 1), OR R3(L,S) AND R4(L,S) (NG = 2)
C
      DO 101 I = 1,NT2
      NO = NBR(I,NG)
      IF(NG.EQ.0)GO TO 101
      L = ID(I)
      LM = L - 1
      IF(L.EQ.0)LM = 1
      BK = BKST(L+1)
      BKM = BKST(LM+1)
      B1 = BIST(L+1)
      B1M = BIST(LM+1)
C
      IF(L.GT.0)GO TO 70
      CL = 1.0
      SL = 0.0
      GO TO 72
      70 CL = STORE(JJJ,2*L-1)
      SL = STORE(JJJ,2*L)
      72 CONTINUE
C
      DO 100 J = 1,NO
      S = IDMAT(I,J,NG)
      JK = S - SC + 2
      BJ = BJST(S+1)
      BJM = -BJST(2)
      IF(S.GT.0)GO TO 80
      CS = 1.0
      SS = 0.0
      GO TO 82
      80 CS = STORE(JJJ,2*S-1)
      SS = STORE(JJJ,2*S)
      BJM = BJST(S)
      82 CONTINUE
C
      CALL INT(SC,RA,RB)
      IF(NG.EQ.2)GO TO 90
      R2(I,JK) = R2(I,JK) + RB*W
      IF(I.GT.1)GO TO 100
      R1(I,JK) = R1(I,JK) + RA*W
      GO TO 100
      90 R3(I,JK) = R3(I,JK) + RB*W
      IF(I.GT.1)GO TO 100
      R4(I,JK) = R4(I,JK) + RA*W
      100 CONTINUE
      101 CONTINUE
C
      GO TO 65
      END
C
      SUBROUTINE MATR(NOP,SC,NO)
C
      INTEGER SC
      DIMENSION R1(10,10),R2(20,10),R3(20,10),R4(10,10)
      DIMENSION RLS(20,20),C(20)
      DIMENSION ALPHA(10,20),BST(31)

```

```

COMMON/A/B,P2,P1,D,NOT,NT2
C  DN/D/R1,R2,R3,R4
C  COMMON/E/R1,R2,C
COMMON/F/ALPHA,BST
C
C  CALCULATE THE COUPLING COEFFICIENTS ALPHA (SC = 1) OR BETA (SC = 2):
C  STORE AS ALPHA(I,J)
C
C  ARG = PI*B*SQRT(P2)*D
C
C  CALCULATE THE K BESSEL FUNCTIONS OF ORDER 0 THROUGH MAX = NOT + NT2,
C  AND STORE THEM WITH INDICES I THROUGH MAX + 1
C
C  MAX = NOT + NT2 + 1
C  I = 0
C  CALL BESKN(ARG,I,1,MAX,BST,J)
C
C  IF(SC.GT.1)GO TO 150
C
C  THIS SECTION CALCULATES ALPHA(K,L)
C  WE NEED THE K-BESSEL FUNCTIONS OF ORDER K = L AND ORDER K + L. FOR THE
C  COSINE SOLUTION, 0 <= K,L <= (NOT-1). TAKING K = I - 1, L = J - 1,
C  WE GET K+L = I + J - 2, K - L = I - J. HOWEVER SINCE THESE START AT
C  ZERO, WE NEED MM = I + J - 1 AND NN = I - J + 1 AS RETRIEVAL INDICES
C  FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
C
C  DO 100 I = 1,NOT
C  EPS = 2.0
C  IF(I.EQ.1)EPS = 1.0
C  DO 100 J = 1,NT2
C  MM = I + J - 1
C  IF(I.GE.J)NN = I - J + 1
C  IF(J.GT.1)NN = J - 1 + 1
C  ALPHA(I,J) = EPS*(BST(MM) + BST(NN))/PI
C  GO TO 200
C
C  100 ALPHA(I,J) = EPS*(BST(MM) + BST(NN))/PI
C  GO TO 200
C
C  THIS SECTION CALCULATES BETA(K,L).
C  WE NEED THE K-BESSEL FUNCTIONS OF ORDER K = L AND K + L. FOR THE SINE
C  EXPANSION, 1 <= K,L <= NOT. TAKING K = I AND L = J SO THAT K + L
C  = I + J AND K - L = I - J, WE NEED MM = I + J + 1 AND NN = I - J + 1
C  AS RETRIEVAL INDICES FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
C
C  NOTE THAT EVEN THOUGH WE CALCULATE BETA(K,L) (SINE SOLUTION), WE STORE
C  THE RESULTS IN ARRAY ALPHA.
C
C  150 CONTINUE
C  DO 170 I = 1,NOT
C  DO 170 J = 1,NT2
C  MM = I + J + 1
C  IF(I.GE.J)NN = I - J + 1
C  IF(J.GT.1)NN = J - 1 + 1
C  ALPHA(I,J) = 2.0*(BST(MM) - BST(NN))/PI
C  170 CONTINUE
C  IF(NOT.GT.1)GO TO 280
C
C  CALL SUBROUTINE CHEBY TO DO THE LINE INTEGRALS TO FIND R1 AND R2
C  R3 AND R4
C
C  CALL CHEBY(NOP,SC)
C
C  CALCULATE THE ELEMENTS OF THE MATRIX FROM THE ELEMENTS OF R1 AND
C  R4, AND SUMS OF ELEMENTS OF R2 AND R3
C
C  280 CONTINUE
C  NN = SC - 1
C  DO 300 I = 1,NOT

```

```

DO 300 J = 1,NOT
  R(I,J) = R(I,I,J)
  RLS(I+NOT,J+NOT) = R4(I,J)
  S2 = 0.0
  S3 = 0.0
DO 295 MM = 1,NT2
  S2 = S2 + ALPHA(I,MM)*R2(MM,J)
  S3 = S3 + ALPHA(I,MM)*R3(MM,J)
295 S3 = S3 + ALPHA(I,MM)*R3(MM,J)
  RLS(I,J+NOT) = S2
300 RLS(I+NOT,J) = S3
  RETURN
END
SUBROUTINE INT(SC,R1,R2)
C
C EVALUATE THE INTEGRANDS OF THE LINE INTEGRALS WHICH BECOME THE
C MATRIX ELEMENTS R1(L,S) THROUGH R4(L,S)
C
  INTEGER S,SC
  COMMON/A/B,P2,P1,D,NOT,NT2
  COMMON/C/P,DMP,RH,DR,CL,SL,CS,SS,BJ,BJM,BK,BKM,BI,BIM,L,S
  XS = 5
  XL = L
  EPS = 1.0
  IF(L.GT.0)EPS = 2.0
  TERM = BK*BJM*DMP + P*BJ*BKM
  TERM = 2.0*B*RH*TERM
  HERM = B*BJM*DMP - P*BJ*BIM
  HERM = P*B*RH*HERM
  IF(L.EQ.S)GO TO 90
  DD = L - S
  TERM = TERM + 2.0*DD*BK*BJ/PI
  HERM = HERM + DD*BI*BJ
90 F = EPS*TERM
  H = EPS*HERM
C
C NOTE THAT I**-(L+1) IS NOT INCLUDED IN F
C NOTE THAT I**L IS NOT INCLUDED IN H
C
  E = 0.0
  G = 0.0
  IF(L.EQ.S)GO TO 100
  E = 2.0*EPS*BK*BJ*DR/(RH*PI)
  G = EPS*BI*BJ*DR/RH
C
C NOTE THAT I**-(L+1) IS NOT INCLUDED IN E
C NOTE THAT I**L IS NOT INCLUDED IN G
C
  100 CONTINUE
C
  IF(SC.EQ.2)GO TO 110
  TERM = XS*CL*SS - XL*SL*CS
  R1 = 4.0*(F*CL*CS + E*TERM)
  R2 = 4.0*(H*CL*CS + G*TERM)
  RETURN
C
110 TERM = XS*SL*CS - XL*SS*CL
  R1 = 4.0*(F*SL*SS - E*TERM)
  R2 = 4.0*(H*SL*SS - G*TERM)
  RETURN
END
SUBROUTINE MATINV(A,N,B,DETER,M)
C
  DIMENSION A(20,20),B(20,1),INDEX(20,3)
  10 DETER = 1.0
  15 DO 20 J = 1,N

```

```

20 INDEX(J,3) = 0
30 IF .50 I = 1,N
C SEARCH FOR PIVOT ELEMENT
C
40 AMAX = 0.
45 DO 105 J = 1,N
IF (INDEX(J,3)-1) 60,105,60
60 DO 100 K = 1,N
IF (INDEX(K,3)-1) 80,100,740
80 IF (AMAX-ABS(A(J,K))) 85,100,100
85 IROW = J
90 ICOLUM = K
AMAX = ABS(A(J,K))
100 CONTINUE
105 CONTINUE
IF (AMAX.EQ.0.0) GO TO 750
INDEX(ICOLUM,3) = INDEX(ICOLUM,3) + 1
260 INDEX(I,1) = IROW
270 INDEX(I,2) = ICOLUM
C INTERCHANGE ROWS TO PUT PIVOT ELEM. IN DIAG.
C
130 IF (IROW-ICOLUM) 140,310,140
140 DETER = -DETER
150 DO 200 L = 1,N
160 SWAP = A(IROW,L)
170 A(IROW,L) = A(ICOLUM,L)
200 A(ICOLUM,L) = SWAP
IF (M) 310,310,210
210 DO 250 L = 1,M
220 SWAP = B(IROW,L)
230 B(IROW,L) = B(ICOLUM,L)
250 B(ICOLUM,L) = SWAP
C DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
310 PIVOT = A(ICOLUM,ICOLUM)
DETER = DETER*PIVOT
330 A(ICOLUM,ICOLUM) = 1.0
340 DO 350 L = 1,N
350 A(ICOLUM,L) = A(ICOLUM,L)/PIVOT
355 IF (M) 360,380,360
360 DO 370 L = 1,M
370 B(ICOLUM,L) = B(ICOLUM,L)/PIVOT
C REDUCE NON PIVOTS ROWS
C
380 DO 550 LI = 1,N
390 IF (LI-ICOLUM) 400,550,400
400 T = A(LI,ICOLUM)
420 A(LI,ICOLUM) = 0.
430 DO 450 L = 1,N
450 A(LI,L) = A(LI,L)-A(ICOLUM,L)*T
455 IF (M) 550,550,460
460 DO 500 L = 1,M
500 B(LI,L) = B(LI,L)-B(ICOLUM,L)*T
550 CONTINUE
C INTERCHANGE COLUMNS
C
600 DO 710 I = 1,N
610 L = N + 1 - I
620 IF (INDEX(L,1)-INDEX(L,2)) 630,710,630
630 JROW = INDEX(L,1)
640 JCOLUM = INDEX(L,2)

```


Program Listing

- LIDGS -

(Reference: Vol. I, Section II.3.5)


```

23  *SHAPE*.20X,B*.6X,N*.6X,D*.6X,B*.8X,9X,ROOTS (P2),.*)
117  IAT/(.1X,R9,12,12,2X,6,F7,2,F8,2,F7,2,F8,3,9F9,4,/.53( '9.4)
118  FUMAT(3X,PMAX,*,F6,3,10X,PMIN,*,F6,3,/)
119  FUMAT(3X,*SHAPE*.20X,R*.6X,N*.6X,D*.6X,B*.8X,9X,
1  *COEFFICIENTS .....*)
120  FUMAT(3X,9F9,4,/.52X,2F9,4)
121  FUMAT(, NOP = NUMBER OF INTEGRATION POINTS =,13)
122  FUMAT(4X,6(A2,C(,12,*) =,E12,5),/4X,4(A2,C(,12,*) =,
1E12,5))
123  FUMAT(1X,A4,*SYMMETRIC CASE:*,14,* ROOT(S) IDENTIFIED*)
124  FUMAT(1X,A3,1X,10E12,4, 6X,10E12,4,/.7X,10E12,4)
125  FUMAT(1X,A3,1X,10E12,4, 6X,10E12,4,/.7X,10E12,4)
126  FUMAT(1X,A3,1X,10E12,4, 6X,10E12,4,/.7X,10E12,4)
127  FUMAT(, NT2 = MAX VAL OF INDEX L IN R2(L,5) =,13)
128  FUMAT(16,F10,6,8E14,7)
129  FUMAT(1X,A3,1X,8E16,8,/.6X,8E16,8,/.7X,8E16,8,/.8X,8E16,8)
130  FUMAT(1X,A3,1X,8E16,8,/.6X,8E16,8,/.7X,8E16,8,/.8X,8E16,8)
1  * NEW FN EVALS: DET =,E15,8)
131  FUMAT(1X,R9,12,13,2X,6,F7,2,F8,2,F7,2,F8,3,9F9,7,/.53X,2F9,7)
132  FUMAT(2X,D =,F5,2,8X,7(F5,2,9X))
133  FUMAT(, IDIAG = DIAGNOSIS CODE, FOR ALL CASES =,13)
134  FUMAT(, $$$SET NBR AND IOWAT FOR I =,13,*, $$$SET)
135  FUMAT(1X,12,*, IDENTICAL GUIDES EQUALLY SPACED IN A ROW*)

```

LIST OF VARIABLES WITH SINGLE ASSIGNED PURPOSES:

```

C ACC  ACCURACY TO WITHIN WHICH THE ROOTS ARE FOUND
C ALPHA  ARRAY OF COUPLING COEFFICIENTS ALPHA (SC = 1) OR BETA (SC = 2)
C ASY  ARRAY OF COEFFICIENTS WHOSE DETERMINANT IS TO BE FOUND (ANTISYMM)
C B  NORMALIZED FREQUENCY (READ IN)
C C  VECTOR OF COEFFICIENTS IN THE FIELD EXPANSION (DESIRED RESULT)
C CFA  ARRAY FOR STORING VALUES OF C FOR DIFFERENT CASES (ANTISYMM CASES)
C CFS  ARRAY FOR STORING VALUES OF C FOR DIFFERENT CASES (SYMMETRIC CASES)
C D  DISTANCE SEPARATING CENTERS OF TWO GUIDES (UNITS OF SEMIMINOR AXIS)
C DET  VALUE OF DETERMINANT (EQUIVALENCED WITH PHI)
C DK  ACCURACY TO WITHIN WHICH THE J AND I BESSEL FUNCTIONS ARE FOUND
C DRDP  D(RHO)/D(PHI) OF THE GUIDE PERIMETERS (EQUIVALENCED WITH XXX)
C DST  ARRAY FOR STORING DIFFERENT VALUES OF D TO BE USED (READ IN)
C ED  INDICATOR SHOWING WHEN NOG IS EVEN (0) OR ODD (1); FOR USE IN MATR
C IDIAG  DIAGNOSIS CODE, TO PRINT EXTRA RESULTS (*) (READ IN)
C IOWAT  ARRAY FOR STORING VALUES OF S NEEDED WHEN R(L,S) OR T(L,S) IS COMPUTED
C INDEX  ARRAY FOR STORING TABLE INDICES NEAR WHICH ROOTS OCCUR
C IROOT  ARRAY FOR STORING INFORMATION USED IN PRINTING RESULTS AT THE END
C JD  ARRAY FOR STORING VALUES OF L NEEDED WHEN R(L,S) OR T(L,S) IS COMPUTED
C JDEX  COUNTER TO KEEP TRACK OF THE NUMBER OF SEPARATE CASES COMPLETED
C LSC  VALUE OF SC FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
C N  SUPERELLIPSE INDEX (READ IN)
C NBR  ARRAY GIVING NUMBER OF VALUES OF S (IN IOWAT) FOR EACH L (IN JD)
C NO  COUNTER IDENTIFYING WHICH ELEMENT OF DST IS TAKEN FOR D
C NDM  DIMENSION OF SYM AND ASY (= NSM*NOT)
C NL  VALUE OF N FROM PREVIOUS CASE (HELPS AVOID REPEATING CALCULATIONS)
C NMWAX  # OF NONZERO VALUES OF D IN DST (LE-4)
C NOG  # OF GUIDES IN THE LINEAR ARRAY
C NOP  # OF INTEGRATION POINTS (READ IN)
C NOT  # OF TERMS IN EXPANSIONS (READ IN)
C NP2  # OF VALUES OF P2 TAKEN TO CONSTRUCT TABLE (READ IN)
C NT2  HIGHEST ALLOWED VALUE OF INDEX L USED IN R2(L,5)
C NR  # OF ROOTS OF DET(ASY)
C NRS  # OF ROOTS OF DET(SYM)
C NSM  SORT OF # OF SUBMATRICES IN SYM AND ASY
C PHI  ANGLE IN (RHO,PHI) COORDINATE SYSTEM
C PI  PI
C PMAX  LARGEST VALUE OF P2 USED IN TABLE (READ IN)
C PMIN  SMALLEST VALUE OF P2 USED IN TABLE (READ IN)

```



```

DK = 10.0**(-K)
IF OP.GT.200INOP = 200
  I.,NOP.LT.20INOP = 20

PRINT THE FIXED INFORMATION READ FROM THE FIRST DATA CARD

WRITE(6,135)NOP
WRITE(6,103)NOT
WRITE(6,127)NT2
WRITE(6,104)N
WRITE(6,105)I
WRITE(6,108)NP2
WRITE(6,121)NOP
WRITE(6,133)IDIAG
WRITE(6,125)
JDEX = 0

.....
SC = 1 IF COSINE TERMS ARE TO BE USED; 2 IF SINE TERMS ARE DESIRED
D = CENTER SEPARATION OF GUIDES, IN UNITS OF SEMIMINOR AXIS
N = INDEX OF OVOID (1 = ELLIPSE, >30 = SQUARE)
R = ASPECT RATIO
B = ASSUMED VALUE OF THE INDEPENDENT PARAMETER, THE BINDING CONSTANT, B
P2 = THE PROPAGATION CONSTANT, TO BE FOUND
PMAX, PMIN = RANGE OF VALUES OF P2 TO BE SEARCHED FOR ROOTS

ONE DATA CARD CORRESPONDS TO ONE INDIVIDUAL CASE
RETURN TO STATEMENT 1 IMMEDIATELY AFTER EACH CASE IS COMPLETE. A BLANK
CARD READ AT STATEMENT 1 CAUSES RESULTS FOR ALL CASES TO BE DUMPED
(PRINTED) AND CONTROL WILL THEN GO TO STATEMENT 2. A BLANK CARD THERE
TERMINATES THE JOB EXECUTION, SO TWO BLANK CARDS SHOULD FOLLOW THE
LAST DATA CARD. IF A DATA CARD CONTAINS MORE THAN ONE NONZERO VALUE OF
DST (THE GUIDE SPACINGS, D) THE PROGRAM WILL RETURN TO STATEMENT 6 TO
PROCESS THOSE PAST THE FIRST. THIS IS A MORE EFFICIENT PROCEDURE THAN
READING EACH VALUE OF D SEPARATELY FROM OTHERWISE IDENTICAL DATA CARDS.

1 READ(5,102)SC,DST(1),N,R,B,PMAX,PMIN,(DST(I),I=2,4)
  IF(R.LE.0.0)GO TO 999
  D = 2.0*R
  ND = 0

COUNT THE NUMBER, NMAX, OF CASES (DIFFERENT D) ON THE LAST DATA CARD

DO 5 I = 1,4
  IF(DST(I).EQ.0.0)GO TO 6
  IF(DST(I).LT.D)DST(I) = D
  NMAX = I
5 CONTINUE
6 ND = ND + 1
  D = DST(ND)
  CALL SECOND(TO)
  IF(PMAX.EQ.0.0)PMAX = 1.0
  IF(N.LE.0.0IN = 1.0
  IF(B.LE.0.0)B = 1.0
  IF(SC.LE.0)SC = 1
  IF(SC.GT.2)SC = 2
  IF(JDEX.EQ.15)GO TO 999
  JDEX = JDEX + 1

INITIALIZE STORAGE ARRAYS FOR THIS CASE

DO 70 I = 7,10
  DO 69 K = 1,NOM
    CFS(JDEX,I=6,K) = 0.0

```

[illegible]


```

IF(DN.EQ.0.0)GO TO 258
  LST = LST + 1
  CFS(15,4,LST) = DN
258 CONTINUE
259 WRITE(6,129)XXX,(CFS(15,4,MAX).MAX = 1,LST)
260 WRITE(6,114)
C
C PRINT THE FINAL MATRIX: STORE THE ROOT
C
265 WRITE(6,114)
  XXX = 3HASY
  DO 252 J = 1,NDM
    IF(IDIAG.LE.0)WRITE(6,124)XXX,(SYM(J,K),K=1,NDM)
    IF(IDIAG.GE.1)WRITE(6,129)XXX,(SYM(J,K),K=1,NDM)
252 CONTINUE
  ROOT(JDEX,1+6) = P2
C
C STORE THE VECTOR OF EXPANSION COEFFICIENTS
C
DO 270 J = 1,NDM
270 CFS(JDEX,1,J) = C(J)
C
C MAKE A FINAL DETERMINATION OF DET FOR THIS ROOT; PRINT IT AND THE EX-
C PANSION COEFFICIENTS
C
CALL MATINV(SYM,NDM,C,DET,0)
WRITE(6,114)
IF(IDIAG.LE.0)WRITE(6,109)XXX,1,P2,PN,ID,DET
IF(IDIAG.GT.0)WRITE(6,130)XXX,1,P2,PN,ID,DET
WRITE(6,114)
  XXX = 2H
  DO 272 J = 1,NSM
    MM = (J-1)*NOT + 1
    NN = J*NOT
    ID = J + SC - 2
272 WRITE(6,122)((XXX,ID,C(K)),K=MM,NN)
  WRITE(6,114)
  IF(DET.EQ.0.0)WRITE(6,113)
250 CONTINUE
C
C *****
C PROCESS OF FINDING SYMMETRIC MODES IS COMPLETE; NOW CONSIDER
C ANTISYMMETRIC MODES: FIND THE ROOTS OF DET(ASY) = 0 PRECISELY
C
300 CONTINUE
  IF(NR.GT.0)GO TO 321
  IROOT(JDEX,4) = 0
  WRITE(6,125)
  IF(ND-NMAX)6,1,1
321 CONTINUE
  IROOT(JDEX,4) = NR
C
C LOOP DO 350... IS EXECUTED ONCE FOR EACH ROOT (I) IDENTIFIED ABOVE
C SUBROUTINE ROOT LOCATES EACH ROOT PRECISELY
C
DO 350 I = 1,NR
  M = INDEX(I,2)
  CALL RTT(TABLE,ACC,PN,NOP,SC,3,M,ID,IDIAG)
C
C PRINT FINAL MATRIX: STORE ROOT
C
  XXX = 3HASY
  DO 352 J = 1,NDM
    IF(IDIAG.LE.0)WRITE(6,124)XXX,(ASY(J,K),K=1,NDM)
    IF(IDIAG.GE.1)WRITE(6,129)XXX,(ASY(J,K),K=1,NDM)
352 CONTINUE

```



```

IF (EQ.EQ.1) NN = 2
D 04 K = NN, NSM
D 04 M = 1, NOT
DO 903 ID = 1, MAX
J = (NSM - K) * NOT + M
903 C(ID) = CFS(I, ID, J) * (-1.0) ** (M+1)
904 WRITE(6, 120) (C(ID), ID=1, MAX)
GO TO 910
905 WRITE(6, 117) ROOT(I, 1), (ROOT(I, M), M=2, 3), (ROOT(I, J), J=2, 6)
910 CONTINUE
C
WRITE(6, 111)
911 XXX = 4HANTI
WRITE(6, 116) PN, NOG, XXX
DO 920 I = 1, JDEX
MAX = IROOT(I, 4)
IF (MAX.EQ.0) GO TO 915
IF (IOIAG.LE.0) WRITE(6, 117) ROOT(I, 1), (ROOT(I, M), M=2, 3),
1(ROOT(I, J), J=2, 6), (ROOTS(I, K), K=1, MAX)
IF (IOIAG.GT.0) WRITE(6, 131) ROOT(I, 1), (ROOT(I, M), M=2, 3),
1(ROOT(I, J), J=2, 6), (ROOTS(I, K), K=1, MAX)
GO TO 920
915 WRITE(6, 117) ROOT(I, 1), (ROOT(I, M), M=2, 3), (ROOT(I, J), J=2, 6)
920 CONTINUE
WRITE(6, 114)
WRITE(6, 114)
WRITE(6, 114)
WRITE(6, 114)
WRITE(6, 119)
DO 930 I = 1, JDEX
MAX = IROOT(I, 4)
IF (MAX.EQ.0) GO TO 925
WRITE(6, 117) ROOT(I, 1), (ROOT(I, M), M=2, 3), (ROOT(I, J), J=2, 6),
1(CFA(I, ID, 1), ID=1, MAX)
DO 922 K = 2, NDM
922 WRITE(6, 120) (CFA(I, ID, K), ID=1, MAX)
NN = 1
IF (EQ.EQ.1) NN = 2
DO 924 K = NN, NSM
DO 921 M = 1, NOT
DO 923 ID = 1, MAX
J = (NSM - K) * NOT + M
923 C(ID) = CFA(I, ID, J) * (-1.0) ** (M)
924 WRITE(6, 120) (C(ID), ID=1, MAX)
GO TO 930
925 WRITE(6, 117) ROOT(I, 1), (ROOT(I, M), M=2, 3), (ROOT(I, J), J=2, 6)
930 CONTINUE
GO TO 2
END
SUBROUTINE REL(SC)
C
C SUBROUTINE REL IDENTIFIES THOSE MATRIX ELEMENTS R(L,S) (COSINE EX-
C PANSION, SC = 1) OR I(L,S) (SINE EXPANSION, SC = 2) WHICH ARE NONZERO.
C THE VALUES OF THE ROW INDICES, L, ARE STORED IN VECTOR ID(I), AND
C THE VALUES OF THE INDICES, S, NBR(I) IN NUMBER, ARE THE ROWS OF
C MATRIX IDMAT(I,J). FOR EXAMPLE, FOR A FIVE-TERM COSINE EXPANSION
C WITH R = 1, THE NONZERO ELEMENTS ARE R(0,0), R(0,4)
C R(1,1), R(1,3)
C R(2,2)
C R(3,1), R(3,3)
C R(4,0), R(4,4)
C
C AND ID(I) = 0 IDMAT(I,J) = 0, 4 1, 3 NBR(I) = 2
C 1 2 2 1
C 2 3 1, 3 2
C 3 4 0, 4 2
C 4

```



```

C      A      N
      INTEGER SC,EO
      DIMENSION STORE(200,42),IDMAT(20,10),ID(20),NBR(20)
      DIMENSION R1(10,10),R2(20,10)
      COMMON/A/B,P2,P1,R,N,D,DK,NDM,NOT,NSM,NT2,EO
      COMMON/B/STORE,IDMAT,ID,NBR
      COMMON/D/R1,R2

C      DO 50 I = 1,NT2
      DO 50 J = 1,NOT
      50 R2(I,J) = 0.0
      DO 55 I = 1,NOT
      DO 55 J = 1,NOT
      55 R1(I,J) = 0.0

C      DO 60 I = 1,NT2
      60 ID(I) = I + SC - 2

C      C      FIND INDICES FOR A CIRCULAR GUIDE
C      IF(N.NE.1.0.OR.R.NE.1.0)GO TO 70
      DO 62 I = 1,NT2
      NBR(I) = 1
      62 IDMAT(I,1) = I + SC - 2
      IF(NT2.EQ.NOT)GO TO 250
      II = NOT + 1
      DO 68 I = 1,NT2
      68 NBR(I) = 0
      GO TO 250

C      70 NOE = 2
      IF(R.EQ.1.0)NDE = 4
      IF(SC.GT.1)GO TO 150

C      C      COSINE EXPANSIONS; HIGHEST INDEX USED = MAX = NOT - 1
C      MAX = NOT - 1
      DO 100 I = 1,NT2
      II = I - 1
      JZ = MOD(II,4) + 1
      NO = 0
      GO TO (85,79,89,79),JZ

C      C      IDENTIFY ELEMENTS OF ODD ROWS (ALWAYS 1,3,5,...)
C      79 J = -1
      80 J = J + 2
      IF(J.GT.MAX)GO TO 100
      NO = NO + 1
      IDMAT(I,NO) = J
      GO TO 80

C      C      IDENTIFY ELEMENTS OF EVEN ROWS
C      85 IF(R.NE.1.0)GO TO 89
      C      ELEMENTS ARE 0,4,8,...
      J = -4
      GO TO 90

C      C      ELEMENTS ARE 0,2,4,... OR 2,6,10...
C      89 J = -2
      90 J = J + NOE

```

```

      I = J.GT.MAX)GO TO 100
      NO = NO + 1
      IDMAT(I,NO) = J
      GO TO 90
100 NBR(I) = NO
      GO TO 250
C
C   SINE EXPANSIONS: HIGHEST INDEX USED = MAX = NOT
C
150 MAX = NOT
      DO 200 I = 1,NT2
      I1 = I
      JZ = MOD(I1,4) + 1
      NO = 0
      GO TO(189,179,185,179),JZ
C
C   IDENTIFY ELEMENTS OF ODD ROWS (ALWAYS 1,3,5,...)
C
179 J = -1
180 J = J + 2
      IF(J.GT.MAX)GO TO 200
      NO = NO + 1
      IDMAT(I,NO) = J
      GO TO 180
C
C   IDENTIFY ELEMENTS OF EVEN ROWS
C
185 IF(R.NE.1.0)GO TO 189
C
C   ELEMENTS ARE 4,8,12,...
C
      J = -2
      GO TO 190
189 J = 0
C
C   ELEMENTS ARE 2,4,6... OR 2,6,10,...
C
190 J = J + NDE
      IF(J.GT.MAX)GO TO 200
      NO = NO + 1
      IDMAT(I,NO) = J
      GO TO 190
200 NBR(I) = NO
250 CONTINUE
C
      WRITE(6,300)
      MAX = 1H
      JZ = 11 - SC
      IF(NT2.LT.JZ)JZ = NT2
      DO 290 I = 1,JZ
      NO = NBR(I)
290 WRITE(6,301)NO,((MAX,ID(I),IDMAT(I,J)),J=1,NO)
C
      IF(JZ.EQ.NT2)RETURN
      JZ = JZ + 1
      DO 295 I = JZ,NT2
      NO = NBR(I)
295 WRITE(6,302)NO,((MAX,ID(I),IDMAT(I,J)),J=1,NO)
C
300 FORMAT(/,*,TABLE OF INDICES (L,S) FOR REQUIRED *,
1*,R(L,S),*)
301 FORMAT(16,3X,10(A1,*,*,11,*,*,11,*,*,3X))
302 FORMAT(16,2X,10(A1,*,*,12,*,*,11,*,*,2X))
C
      RETURN
      END

```

```

C      SUBROUTINE CODE(N,R,NL,RL,SC,LSC,LST)
C      4 - N,NL
C      INTEGER SC
C      FIND LST: CODE FOR DECIDING WHEN TO RECALCULATE INDICES (2),
C      PERIMETER (3), OR BOTH (4).
C
C      LST = 1
C      IF(R.EQ.RL)GO TO 71
C
C      FIND LST FOR CASES WITH R.NE.RL
C
C      LST = 3
C      IF(R.NE.1.0.AND.RL.NE.1.0)GO TO 73
C      GO TO 72
C
C      71 IF(N.EQ.NL)GO TO 73
C
C      FIND LST FOR CASES WITH R.EQ.RL
C
C      LST = 3
C      IF(R.NE.1.0)GO TO 73
C      IF(N.NE.1.0.AND.NL.NE.1.0)GO TO 73
C      72 LST = 4
C      GO TO 74
C
C      73 IF(SC.EQ.LSC)GO TO 74
C
C      CHANGE LST WHEN SC.NE.LSC
C
C      LST = LST + 1
C      74 LSC = SC
C      RL = R
C      NL = N
C      RETURN
C      END
C
C      SUBROUTINE RTT(TABLE,ACC,PN,NOP,SC,JJ,M,ID,IDIAG)
C
C      SUBROUTINE ROOT PRECISELY LOCATES THE ROOT SPECIFIED BY M AND
C      JJ (USING ELEMENTS M AND M+1 IN THE ARRAY, TABLE(M,JJ)) AND THEN
C      DETERMINES THE FIELD EXPANSION COEFFICIENTS C(J) CORRESPONDING TO
C      THAT ROOT
C
C      REAL N
C      INTEGER SC,EO
C      DIMENSION SYM(30,30),ASY(30,30),C(30)
C      DIMENSION TABLE(51,9)
C      DIMENSION SIN(30,30)
C      COMMON/A/B,P2,P1,R,N,D,DK,NDM,NOT,NSM,NT2,EO
C      COMMON/E/SYM,ASY,C
C
C      700 FORMAT(/, 'SEARCH ID= 7X, *X0*, 11X, *X1*, 11X, *X2*, 10X, *F0*, 9X, *F1*,
C      19X, *F2*, 11X, *X*, 10X, *F*, 10X, *DX*')
C      701 FORMAT(/, ' $$$$ ERROR IN HINTS OR MULLER, IER =, 13, $$$$$$')
C
C      IF(IDIAG.GE.3)WRITE(6,700)
C      IER = 0
C      ID = 0
C      IF(M.LT.100)GO TO 230
C      P2 = TABLE(M-100,1)
C      CALL MATR(NOP,SC,1)
C      GO TO 248
C
C      HALF-INTERVAL SEARCH
C

```

```

230 G1 = TABLE(M+1,JJ)
C  G1 = TABLE(M,JJ)
C  X1 = TABLE(M+1,1)
C  X2 = TABLE(M,1)
C  DN = X2 - X1
C  K = 0
C
C  IF G1 AND G2 ARE TWO OR MORE ORDERS OF MAGNITUDE DIFFERENT, USE
C  A HALF-INTERVAL SEARCH TO NARROW THE RANGE X1 TO X2
C
235 W = -G1/G2
C  IF (WM.LT.100..AND.WM.GT..01)GO TO 240
C  IF (DN.LT.ACC)GO TO 240
C  DN = DN/3.0
C  CALL HINTS(X1,X2,G1,G2,DN,PN,NOP,SC,JJ,ID,IDIAG,IER)
C  IF (IER.GT.0)WRITE(6,701)IER
C  K = K + 1
C  IF (PN.EQ.0.0)GO TO 248
C  GO TO 235
C
C  MULLER'S METHOD TO GET THE ROOT PRECISELY
C
240 CALL MULLER(X1,X2,G1,G2,ACC,PN,NOP,SC,JJ,ID,IDIAG,IER)
C
C  ID = ID + K
248 CONTINUE
C  IF (IER.GT.0)WRITE(6,701)IER
C
C  CALCULATE CFS, THE RATIO OF COEFFICIENTS C3, C3,... TO C1, OR
C  D3, D3,... TO D1
C
C  IF (JJ.EQ.3)GO TO 260
C  CALL LOAD(SYM,SINV,C,JJ,SC,M)
C  GO TO 266
C
260 CONTINUE
C  CALL LOAD(ASY,SINV,C,JJ,SC,M)
266 LST = NDM - 1
C
C  CALL MATINV(SINV,LST,C,G,1)
C
C  STACK THE COEFFICIENTS IN THE PROPER ORDER IN VECTOR C, INCLUDING 1.000
C
C  IF (M.EQ.NDM)GO TO 280
C  DO 270 J = M,LST
C  K = NDM + M - J
270 C(K) = C(K-1)
280 C(M) = 1.0
C
C  CHECK THE NORMALIZATION OF THE COEFFICIENTS
C
C  IF (NSM.EQ.1)RETURN
C  LST = MOD(M,MOT)
C  G = 0.0
C  DO 290 K = 1,NSM
C  DN = ABS(C(LST+K-1))
C  IF (DN.LT.G)GO TO 290
C  J = LST + K - 1
C  G = DN
290 CONTINUE
C  IF (J.EQ.M)RETURN
C  G = C(J)
C  DO 295 K = 1,NDM
295 C(K) = C(K)/G
C  RETURN

```

```

END
S( ROUTINE MULLER(X0,X2,F0,F2,ACC,PN,NOP,SC,JJ,IO,IDIAG,IER)
MULLER'S ALGORITHM FOR THE SOLUTION OF DET(RLS) = 0
INVERSE PARABOLIC INTERPOLATION USING POINTS
(X0,F0), (X1,F1), AND (X2,F2)
TO FIND THE NEW POINT (X,F), WHERE X0 < X1 < X2 AND F0*F2 < 0
ALWAYS HOLD. SIGN IS THE SIGN OF F0, AND DOES NOT CHANGE FROM ITS
INITIAL VALUE. THE TWO ROOTS OF THE QUADRATIC NECESSARILY LIE
BETWEEN X0 AND X1, AND BEYOND X0 OR X2 IF (A) SIGN*F1 < 0
BETWEEN X1 AND X2, AND BEYOND X0 OR X2 IF (B) SIGN*F1 > 0
(THAT IS, FOR EXAMPLE, IN CASE (A) THERE IS NO ROOT BETWEEN X1 AND X2.)
CHOOSE THE ROOT INSIDE THE INTERVAL.
XL IS THE LARGEST X FOR WHICH F*SIGN > 0
XU IS THE SMALLEST X FOR WHICH F*SIGN < 0
INTEGER SC,EO
REAL N
DIMENSION SYM(30,30),ASY(30,30),C(30)
COMMON/A,B,P2,PI,R,N,D,DM,NDM,NOT,NSM,NT2,EO
COMMON/E/SYM,ASY,C
C
ID = 0
IER = 0
IF(X2.LE.X0)GO TO 290
PN = X2 - X0
W = ALOG(PN/ACC)/-.693147
LST = W + 1.5
C
LST GIVES THE NUMBER OF ITERATIONS WHICH A HALF-INTERVAL SEARCH
WOULD REQUIRE FOR THESE INITIAL VALUES AND DESIRED ACCURACY. IF
MULLER'S ALGORITHM EXCEEDS THIS, EXIT WITH ERROR CONDITION 3.
700 FORMAT(' MULLR:',I3,3F13.10,3E11.4,F13.10,2E11.4)
702 FORMAT(' MULLR:',I3,72X,F13.10,E11.4)
C
ASSUME THAT THE INITIAL VALUES ARE SUCH THAT X0 < X2
IF NOT, RETURN (IER = 4)
XL = X0
XU = X2
SIGN = F0/ABS(F0)
C
SPLIT THE INTERVAL AND EVALUATE THE DETERMINANT AT THE MIDPOINT
40 X = (X0 + X2)/2.0
P2 = X
CALL MATR(NOP,SC,1)
IF(JJ.EQ.2)CALL MATINV(SYM,NDM,C,F,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NDM,C,F,0)
ID = ID + 1
IF(F*SIGN)50,200,60
50 XU = X
GO TO 100
60 XL = X
C
EVALUATE THE NEW X FROM THE OLD X0, X1, AND X2
100 X1 = X
F1 = F
DD01 = (F1 - F0)/(X1 - X0)
DD12 = (F2 - F1)/(X2 - X1)
DD012 = (DD12 - DD01)/(X2 - X0)
W = DD01 + (X1 - X0)*DD012

```

```

C
U = DD012/M
G = F1/M
I
U = 1.0 - 4.0*GL
IF(U.LT.0.0)GO TO 260
U = 2.0*GL/(1.0 + SORT(U))
X = X1 - U
IF(X.GT.X0.AND.X.LT.X2)GO TO 145
U = 1.0 - 4.0*GL*DD012/M
U = 2.0*GL/(1.0 - SORT(U))
X = X1 - U
145 CONTINUE
IF(X.GT.1.0.OR.X.LT.0.0)GO TO 270
P2 = X
CALL MATR(NOP,SC,1)
ID = ID + 1
PN = ABS(XU - XL)
IF(PN.LT.ACC)GO TO 210
IF(ID.GT.LST)GO TO 280
IF(JJ.EQ.2)CALL MATINV(SYM,NDM,C,F,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NDM,C,F,0)
C
IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X,F,PN
C
C RESET THE UPPER OR LOWER BOUND ON THE ROOT
C
IF(F*SIGN)150,200,155
150 XU = X
GO TO 160
155 XL = X
C
C CHANGE X0 OR X2 TO THE PREVIOUS X1; THE NEW X1 IS X
C
160 IF(F1*SIGN)165,200,170
165 X2 = X1
F2 = F1
GO TO 175
170 X0 = X1
F0 = F1
175 GL = ABS(U)
IF((F*F1).LT.0.0)GO TO 100
IF(GL.GT.ACC)GO TO 100
C
C IF POINTS ACCUMULATE CLOSELY ON ONE SIDE OF THE TRUE ROOT, A SINGLE
C SMALL JUMP MAY SPAN THE ROOT. THEREFORE CHANGE X BY A LITTLE LESS
C THAN ACC AND EVALUATE THE DETERMINANT. IF THIS FAILS (THAT IS, IF THE
C FUNCTION HAS THE SAME SIGN AGAIN) SPLIT THE INTERVAL IN HALF AND
C CONTINUE.
C
C
C LINEAR EXTRAPOLATION PAST THE ANTICIPATED ZERO
C
W = 1.0E-12
IF(X.NE.X1)GO TO 180
XT = W
IF((F*SIGN).GT.0.0)XT = -W
GO TO 185
180 XT = 2.0*F*(X1 - X)/(F1 - F)
GL = ABS(XT)
IF(GL.GT.W)GO TO 185
XT = W*XT/GL
185 XT = X - XT
P2 = XT
CALL MATR(NOP,SC,1)
IF(JJ.EQ.2)CALL MATINV(SYM,NDM,C,FT,0)
IF(JJ.EQ.3)CALL MATINV(ASY,NDM,C,FT,0)

```

```

10 = ID + 1
{  DIAG.GE.3)WRITE(6,702)ID,X1,FT
  I,FT,F)190,200,40
190 IF(FT-SIGN)192,200,193
192 XU = XT
   X2 = XT
   F2 = FT
   GO TO 100
193 XL = XT
   X0 = XT
   F0 = FT
   GO TO 100

C  IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH MAY HAVE BEEN
C  ALTERED IN MATINV) IS RECOMPUTED.
C  200 PN = 0.0
C  CALL MATR(NOP.SC,1)
   ID = ID + 1
C  NORMAL RETURN:
C  210 IF(IDIAG.GE.3)WRITE(6,700)ID,X0,X1,X2,F0,F1,F2,X
   RETURN
C  ERROR RETURN 1: COMPLEX ROOT
C  260 IER = 1
   RETURN
C  ERROR RETURN 2: PROCEDURE IS OSCILLATING
C  270 IER = 2
   RETURN
C  ERROR RETURN 3: MORE ITERATIONS THAN HALF-INTERVAL SEARCH WOULD REQUIRE
C  280 IER = 3
   GO TO 210
C  ERROR RETURN 4: X0 > X2 INITIALLY
C  290 IER = 4
   RETURN
END
SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,JJ,ID,IDIAG,IER)
C  SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE THE ROOTS OF
C  THE EQUATION DET(RLS) = 0 OR DET(TLS) = 0 TO WITHIN A DESIRED ACCURACY
C  ACC. ID IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
C  INTEGER SC,EO
C  REAL N
C  DIMENSION SYM(30,30),ASY(30,30),C(30)
C  COMMON/A/B,P2,P1,R,N,D,DK,NDM,NOT,NSM,NT2,EO
C  COMMON/E/SYM,ASY,C
C  700 FORMAT(* HINTS:*,I3,I3X,2F13.10,11X,2E11.4,24X,E11.4)
C  ID = 0
   IER = 0
C  100 PN = ABS(X2 - X1)
C  IF(IDIAG.GE.3)WRITE(6,700)ID,X1,X2,G1,G2,PN
   IF(PN.LT.ACC)RETURN

```

```

ID = ID + 1
IF (ID.GT.25) GO TO 230
  (X1 + X2)/2.0
  CALL MATR(NOP,SC,1)
  IF (JQ.EQ.2) CALL MATINV(SYM,NOM,C,G,0)
  IF (JQ.EQ.3) CALL MATINV(ASY,NOM,C,G,0)
  IF (G.GT.1) 120,210,130
120 X2 = P2
    G2 = G
    GO TO 100
130 X1 = P2
    G1 = G
    GO TO 100
210 PN = 0.0
    RETURN
C
C ERROR RETURN: TOO MANY ITERATIONS
C
230 IER = 6
    RETURN
    END
    SUBROUTINE LOAD(SS,SINV,C,MM,SC,ID)
C
C SUBROUTINE LOAD LOADS SS (SEE SYM OR ASY) INTO ARRAY SINV, ELIMINATING
C THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
C IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
C LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
C INATED. THUS, SINV IS (NDM-1) BY (NDM-1) IN DIMENSION, AND C IS A
C VECTOR (NDM-1) IN LENGTH.
C
C ONLY THE FIRST THREE ELEMENTS OF EACH SECTION OF THE DIAGONAL ARE
C SEARCHED FOR THE SMALLEST VALUE.
C
C INTEGER SC,EO
C REAL N
C DIMENSION SS(30,30),SINV(30,30),C(30)
C COMMON/A/B,P2,P1,R,N,D,DK,NDM,NOT,NSM,NT2,EO
C
C FIRST FIND THE SMALLEST DIAGONAL ELEMENT
C
G = 1.0E+90
ID = 1
K = NOM
IF (K.GT.3) K = 3
DO 200 I = 1,NSM
  DO 200 M = 1,K
    J = (I-1)*NOT + M
    H = ABS(SS(J,J))
    IF (H.GT.G) GO TO 200
    G = H
    KK = ID
    ID = J
  200 CONTINUE
C
C PRECAUTION: AVOID CHOOSING FOR ID A NUMBER SUCH THAT C(ID) IS IDEN-
C TICAL ZERO. THIS CAN HAPPEN WITH AN ODD NUMBER OF GUIDES IF C(ID)
C IS FROM THE FIELD EXPANSION OF THE MIDDLE GUIDE. IF IT DOES, USE
C THE SECOND SMALLEST DIAGONAL ELEMENT (KK) FOR ID
C
IF (EO.EQ.0) GO TO 250
JJ = NOM - NOT
IF (ID.LE.JJ) GO TO 250
G = (-1.0)**(ID-JJ+SC)
H = (-1.0)**(MM+SC+1)

```



```

C IF G > 0. WE HAVE SELECTED AN EVEN COEFFICIENT
C I < 0. WE HAVE SELECTED AN ODD COEFFICIENT
C I > 0. WE WANT AN EVEN COEFFICIENT
C IF H < 0. WE WANT AN ODD COEFFICIENT
C IF((G*H).GT.0.0)GO TO 250
C ID = KK
C ELIMINATE THE ROW AND COLUMN IDENTIFIED BY ID
C
250 JJ = 0
DO 300 J = 1,NDM
IF(J.EQ.ID)GO TO 300
JJ = JJ + 1
C(JJ) = -SS(J,ID)
KK = 0
DO 299 K = 1,NDM
IF(K.EQ.ID)GO TO 299
KK = KK + 1
SINV(JJ,KK) = SS(J,K)
299 CONTINUE
300 RETURN
END
SUBROUTINE CHEBY(NOP,SC)
C
INTEGER S,SC,SMIN,SMAX,E0
REAL N
DIMENSION STORE(200,42),IDMAT(20,10),ID(20),NBR(20)
DIMENSION R1(10,10),R2(20,10)
DIMENSION BJST(11),B1ST(21),BKST(31)
COMMON/A/B,P2,PI,R,N,D,DK,NDM,NOT,MSM,NT2,E0
COMMON/B/STORE,IDMAT,ID,NBR
COMMON/C/RH,DR,CL,SL,CS,SS,BJ,BK,BKM,BT,BIM,L,S
COMMON/D/R1,R2
C
CHEBYSHEV INTEGRATION TO PRODUCE R1(L,S) AND R2(L,S) (SC = 1)
OR T1(L,S) AND T2(L,S) (SC = 2)
C
PTS = NOP
W = PI/(PTS+2.0)
DO 50 I = 1,NT2
NO = NBR(I)
DO 50 J = 1,NO
JK = IDMAT(I,J) - SC + 2
R1(I,JK) = 0.0
R2(I,JK) = 0.0
LMAX = ID(NT2)
SMAX = ID(NOT)
LMIN = 0
SMIN = 0
50 R2(I,JK) = 0.0
C
LOOP DO 100 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
C LOOP DO 100 I... CORRESPONDS TO L
C LOOP DO 100 J... CORRESPONDS TO S
C
DO 100 JJJ = 1,NOP
RH = STORE(JJJ,1)
DR = STORE(JJJ,2)
C
C CALCULATE AND STORE THE J BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT
C J0 IS STORED AS BJST(1), J1 IS STORED AS BJST(2), ETC.
C
ARG = PI*B*RH*SQRT(1.0 - P2)
CALL JRECUR(ARG,SMIN,SMAX,BJST,DK)
C

```

```

C      CALCULATE AND STORE THE I BESSEL FUNCTIONS NEEDED BY SUBROUTINE INT.
C      IF 'S STORED AS BIST(1), I1 IS STORED AS BIST(2), ETC.
C      (
C      ARG = PI*B*RH*SQRT(P2)
C      CALL KRECUR(ARG,LMIN,LMAX,BIST,DK)
C
C      CALCULATE AND STORE THE K BESSEL FUNCTIONS NEEDED IN SUBROUTINE INT.
C      K0 IS STORED AS BKST(1), K1 IS STORED AS BKST(2), ETC.
C      CALL KRECUR(ARG,LMAX,BKST,DK)
C
C      CALCULATE R1(L,S) AND R2(L,S) OR T1(L,S) AND T2(L,S)
C      DO 100 I = 1,NT2
C      L = ID(I)
C      LM = L - 1
C      IF(L.EQ.0) LM = 1
C      BK = BKST(L+1)
C      BKM = BKST(LM+1)
C      B1 = BIST(L+1)
C      B1M = BIST(LM+1)
C
C      NO = NBR(I)
C      IF(L.GT.0) GO TO 70
C      CL = 1.0
C      SL = 0.0
C      GO TO 72
C
C      70 CL = STORE(JJJ,2*L+1)
C      SL = STORE(JJJ,2*L+2)
C      72 CONTINUE
C
C      DO 100 J = 1,NO
C      S = IDMAT(I,J)
C      JK = S - SC + 2
C      BJ = BJST(S+1)
C      BJM = -BJST(2)
C      IF(S.GT.0) GO TO 80
C      CS = 1.0
C      SS = 0.0
C      GO TO 82
C
C      80 CS = STORE(JJJ,2*S+1)
C      SS = STORE(JJJ,2*S+2)
C      BJM = BJST(S)
C      82 CONTINUE
C
C      CALL INT(SC,RA,RB)
C      R2(I,JK) = R2(I,JK) + RB*W
C      IF(I.GT.NOT) GO TO 100
C      R1(I,JK) = R1(I,JK) + RA*W
C      100 CONTINUE
C
C      RETURN
C      END
C
C      SUBROUTINE KRECUR(X,NMAX,B,DK)
C
C      SUBROUTINE KRECUR CALCULATES THE MODIFIED BESSEL FUNCTIONS KN(X)
C      OF ORDER N AND ARGUMENT X FOR N BETWEEN 0 AND NMAX. IT CALLS
C      SUBROUTINE BESK TO FIND K0(X) AND K1(X), AND THEN
C      USES A RECURSION RELATION TO CALCULATE THE HIGHER ORDERS. THE
C      VECTOR B CONTAINS THE RESULTANT BESSEL FUNCTIONS, WITH THE VECTOR
C      INDEX BEING ONE GREATER THAN THE ORDER OF THE FUNCTION STORED AT
C      THAT LOCATION. THAT IS, K0 IS STORED AS B(1), K1 AS B(2), ETC.
C      DIMENSION B(31)

```

```

C
1  I MAX.GT.30)NMAX = 30
2  I..4.GT.0.0)GO TO 5
DO 4 N = 1,NMAX
4  B(N+1) = .99E+99
  B(1) = .99E+99
  RETURN
C
5  N = 0
  CALL BESK(X,N,BKM,K)
  B(1) = BKM
  IF(NMAX.EQ.N)RETURN
  N = 1
  CALL BESK(X,N,BK,K)
10 B(N+1) = BK
  IF(N.GE.NMAX)RETURN
  F = N
  BKP = BKM + 2.0*F*BK/X
  BKM = BK
  BK = BKP
  N = N + 1
  GO TO 10
END
SUBROUTINE JRECUR(X,NMIN,NMAX,B,DK)
C
C SUBROUTINE JRECUR CALCULATES THE MODIFIED BESSEL FUNCTIONS JN(X)
C OF ORDER N AND ARGUMENT X FOR N BETWEEN NMIN AND NMAX. IT CALLS
C SUBROUTINE BESJ TO FIND JN(X) FOR N = NMAX AND NMAX - 1, AND THEN
C USES A RECURSION RELATION TO CALCULATE THE LOWER ORDERS. THE
C VECTOR B CONTAINS THE RESULTANT BESSEL FUNCTIONS, WITH THE VECTOR
C INDEX BEING ONE GREATER THAN THE ORDER OF THE FUNCTION STORED AT
C THAT LOCATION. THAT IS, J0 IS STORED AS B(1), J1 AS B(2), ETC.
C
C DIMENSION B(21)
C
  IF(NMIN.LT.0)NMIN = 0
  IF(NMAX.GT.20)NMAX = 20
  IF(NMIN.GT.NMAX)NMIN = NMAX
  IF(X.GT.0.0)GO TO 5
  K = NMIN
  IF(NMIN.EQ.0)K = 1
  DO 4 N = K,NMAX
4  B(N+1) = 0.0
  B(1) = 1.0
  RETURN
C
5  EPS = DK/10.
  IF(EPS.LT.1.0E-14)EPS = 1.0E-14
  CALL BESJ(X,NMAX,BJP,EPS,K)
  B(NMAX+1) = BJP
  IF(NMIN.EQ.NMAX)RETURN
  N = NMAX - 1
  CALL BESJ(X,N,BJ,EPS,K)
10 B(N+1) = BJ
  IF(NMIN.EQ.N)RETURN
  F = N
  BJM = 2.0*F*BJ/X - BJP
  BJP = BJ
  BJ = BJM
  N = N - 1
  GO TO 10
END
SUBROUTINE MATR(NOP,SC,ND)
C
  REAL N
  INTEGER SC,EO
  DIMENSION B(10,10),B9(20,10)

```

```

DIMENSION ALPHA(10,20,5),BST(31),SUM(5)
COMMON/ASYM/ASY,C
COMMON/D/R1,R2
COMMON/E/SYM,ASY,C
COMMON/F/ALPHA
C
C CALCULATE THE COUPLING COEFFICIENTS ALPHA (SC = 1) OR BETA (SC = 2);
C STORE AS ALPHA(I,J,K). THE INDEX K SPECIFIES WHICH PAIR OF GUIDES
C IS CONSIDERED. THAT IS, THE ARGUMENT CONTAINS THE DISTANCE BETWEEN
C GUIDE # 1 AND GUIDE # 2, # 1 AND # 3, ETC. FOR UNIFORM SPACINGS,
C THE NUMBER OF PAIRS REQUIRED IS ONE LESS THAN NOG.
C
NOG = 2*NSM - EO
NS = NOG - 1
DO 200 K = 1,NS
SIGN = K
ARG = PI*B*SORT(P2)*SIGN-D
C
C CALCULATE THE K BESSEL FUNCTIONS OF ORDER 0 THROUGH MAX = NOT + NT2,
C AND STORE THEM WITH INDICES 1 THROUGH MAX + 1
C
MAX = NOT + NT2
CALL KRECUR(ARG,MAX,BST,DK)
C
IF(SC.GT.1)GO TO 150
C
C THIS SECTION CALCULATES ALPHA(K,L)
C WE NEED THE K-BESSEL FUNCTIONS OF ORDER K - L AND ORDER K + L. FOR THE
C COSINE SOLUTION, 0 <= K,L <= (NOT-1). TAKING K = I - 1, L = J - 1,
C WE GET K+L = I + J - 2, K - L = I - J. HOWEVER SINCE THESE START AT
C ZERO, WE NEED MM = I + J - 1 AND NN = I - J + 1 AS RETRIEVAL INDICES
C FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
C
DO 100 I = 1,NOT
EPS = 2.0
IF(I.EQ.1)EPS = 1.0
DO 100 J = 1,NT2
MM = I + J - 1
IF(I.GE.J)NN = I - J + 1
IF(J.GT.1)NN = J - I + 1
100 ALPHA(I,J,K) = EPS*(BST(MM) + BST(NN))/PI
GO TO 200
C
C THIS SECTION CALCULATES BETA(K,L).
C WE NEED THE K-BESSEL FUNCTIONS OF ORDER K - L AND K + L. FOR THE SINE
C EXPANSION, 1 <= K,L <= NOT. TAKING K = I AND L = J SO THAT K + L
C = I + J AND K - L = I - J, WE NEED MM = I + J + 1 AND NN = I - J + 1
C AS RETRIEVAL INDICES FOR THE ARRAY BST CONTAINING THE BESSEL FUNCTIONS.
C
NOTE THAT EVEN THOUGH WE CALCULATE BETA(K,L) (SINE SOLUTION), WE STORE
C THE RESULTS IN ARRAY ALPHA.
C
150 CONTINUE
DO 170 I = 1,NOT
DO 170 J = 1,NT2
MM = I + J + 1
IF(I.GE.J)NN = I - J + 1
IF(J.GT.1)NN = J - I + 1
170 ALPHA(I,J,K) = 2.0*(BST(MM) - BST(NN))/PI
200 CONTINUE
IF(ND.GT.1)GO TO 280
C
C CALL SUBROUTINE CHEBY TO DO THE LINE INTEGRALS TO FIND R1 AND R2
C CALL CHEBY(NOP,SC)

```

```

C      (
C      CALCULATE THE ELEMENTS OF THE MATRIX FROM THE ELEMENTS OF R1 AND
C      SUMS OF ELEMENTS OF R2. FIRST FIND THE (1,1) ELEMENT OF EACH OF THE
C      SUBMATRICES. THE THE (1,2) ELEMENT OF EACH, AND SO ON.
C
C      280 CONTINUE
C      DO 300 I = 1,NOT
C      DO 300 J = 1,NOT
C      DO 285 K = 1,NS
C
C      CALCULATE THE SUMS OF ALPHA*R2 FOR THE (I,J) ELEMENT OF EACH OF
C      THE SUBMATRICES
C      SUM(K) = 0.0
C      DO 285 M = 1,NT2
C      285 SUM(K) = SUM(K) + R2(M,J)*ALPHA(I,M,K)
C
C      CALCULATE THE FIRST TERM IN THE (I,J) ELEMENT OF EACH OF THE SUBMATRICES
C
C      DO 290 K = 1,NSM
C      DO 290 M = 1,NSM
C      MM = (K - 1)*NOT
C      NN = (M - 1)*NOT
C      IF(K-M)/286,287,288
C
C      UPPER-RIGHT SUBMATRICES
C
C      286 SIGN = (-1.0)**(SC+J)
C      IF(M.EQ.NSM.AND.EQ.EQ.1)SIGN = 0.0
C      SYM(I+MM,J+NN) = SIGN*SUM(M-K)
C      ASY(I+MM,J+NN) = SIGN*SUM(M-K)
C      GO TO 290
C
C      DIAGONAL SUBMATRICES
C
C      287 SYM(I+MM,J+NN) = R1(I,J)
C      ASY(I+MM,J+NN) = R1(I,J)
C      GO TO 290
C
C      LOWER-LEFT SUBMATRICES
C
C      288 SIGN = (-1.0)**(SC+I)
C      SYM(I+MM,J+NN) = SIGN*SUM(K-M)
C      ASY(I+MM,J+NN) = SIGN*SUM(K-M)
C      290 CONTINUE
C
C      CALCULATE THE SECOND TERM IN THE (I,J) ELEMENT OF EACH OF THE SUBMATRICES
C      ADD THESE TO THE PREVIOUS RESULTS TO COMPLETE SYM(I,J) AND ASY(I,J).
C
C      SIGN = 1.0
C      IF(SC.EQ.2)SIGN = -1.0
C      DO 295 K = 1,NSM
C      DO 295 M = 1,NSM
C      IF(M.EQ.K.AND.K.EQ.NSM.AND.EQ.EQ.1)GO TO 295
C      MM = (K - 1)*NOT
C      NN = (M - 1)*NOT
C      SYM(I+MM,J+NN) = SYM(I+MM,J+NN) + SIGN*SUM(NOQ-K-M+1)
C      ASY(I+MM,J+NN) = ASY(I+MM,J+NN) - SIGN*SUM(NOQ-K-M+1)
C      295 CONTINUE
C      300 CONTINUE
C      RETURN
C      END
C      SUBROUTINE INT(SC,R1,R2)
C
C      EVALUATE THE INTEGRANDS OF THE LINE INTEGRALS WHICH BECOME THE
C      MATRIV ELEMENTS B1(I, 6) AND B2(I,5)

```


AD-A139 604

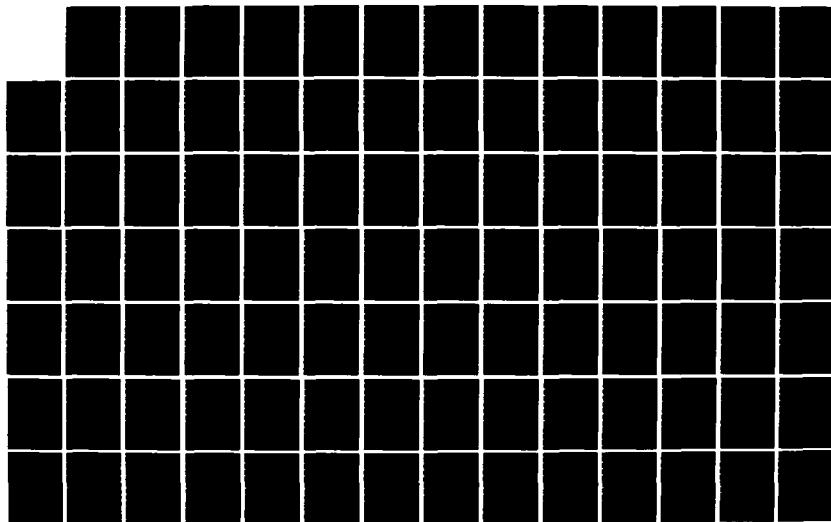
MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND
ELECTRON TRANSPORT VOLUME 2(U) ARCON CORP WALTHAM MA
S WOOLF ET AL. JAN 84 RADC-TR-83-313-VOL-2
F19628-78-C-0188

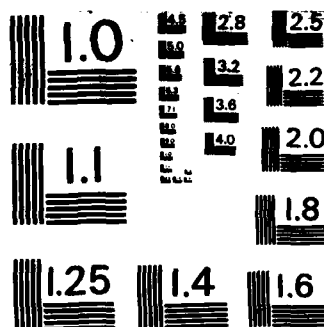
2/5

UNCLASSIFIED

F/G 9/2

NL





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


```

      Y INDEX(K,3)-1) 80,100,740
80 I = MAX-ABS(A(J,K)) 85,100,100
85 IROW = J
90 ICOL = K
  AMAX = ABS(A(J,K))
100 CONTINUE
105 CONTINUE
  INDEX(ICOL,3) = INDEX(ICOL,3) + 1
260 INDEX(I,1) = IROW
270 INDEX(I,2) = ICOL
C
C INTERCHANGE ROWS TO PUT PIVOT ELEM. IN DIAG.
C
  IF(AMAX)130,999,130
130 IF(IROW-ICOL) 140,310,140
140 DETER = -DETER
150 DO 200 L = 1,N
160 SWAP = A(IROW,L)
170 A(IROW,L) = A(ICOL,L)
200 A(ICOL,L) = SWAP
  IF(M) 310,310,210
210 DO 250 L = 1,M
220 SWAP = B(IROW,L)
230 B(IROW,L) = B(ICOL,L)
250 B(ICOL,L) = SWAP
C
C DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
310 PIVOT = A(ICOL,ICOL)
  DETER = DETER*PIVOT
330 A(ICOL,ICOL) = 1.0
340 DO 350 L = 1,N
350 A(ICOL,L) = A(ICOL,L)/PIVOT
355 IF (M) 380,380,360
360 DO 370 L = 1,M
370 B(ICOL,L) = B(ICOL,L)/PIVOT
C
C REDUCE NON PIVOTS ROWS
C
380 DO 550 LI = 1,N
390 IF(LI-ICOL) 400,550,400
400 I = A(LI,ICOL)
420 A(LI,ICOL) = 0.
430 DO 450 L = 1,N
450 A(LI,L) = A(LI,L)-A(ICOL,L)*I
455 IF (M) 550,550,460
460 DO 500 L = 1,M
500 B(LI,L) = B(LI,L)-B(ICOL,L)*I
550 CONTINUE
C
C INTERCHANGE COLUMNS
C
600 DO 710 I = 1,N
610 L = N + 1 - I
620 IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
630 JROW = INDEX(L,1)
640 JCOL = INDEX(L,2)
650 DO 705 K = 1,N
660 SWAP = A(K,JROW)
670 A(K,JROW) = A(K,JCOL)
700 A(K,JCOL) = SWAP
705 CONTINUE
710 RETURN
999 DETER = 0.0
  RETURN

```

```

C
C
C      SUBROUTINE IRECUR(X,NMIN,NMAX,B,OK)
C      SUBROUTINE IRECUR CALCULATES THE MODIFIED BESSEL FUNCTIONS IN(X)
C      OF ORDER N AND ARGUMENT X FOR N BETWEEN NMIN AND NMAX. IT CALLS
C      SUBROUTINE IBESS TO FIND IN(X) FOR N = NMAX AND NMAX - 1, AND THEN
C      USES A RECURSION RELATION TO CALCULATE THE LOWER ORDERS. THE
C      VECTOR B CONTAINS THE RESULTANT BESSEL FUNCTIONS, WITH THE VECTOR
C      INDEX BEING ONE GREATER THAN THE ORDER OF THE FUNCTION STORED AT
C      THAT LOCATION.
C
C      DIMENSION B(21)
C
C      IF(NMIN.LT.0)NMIN = 0
C      IF(NMAX.GT.20)NMAX = 20
C      IF(NMIN.GT.NMAX)NMIN = NMAX
C      IF(X.GT.0.0)GO TO 5
C      K = NMIN
C      IF(NMIN.EQ.0)K = 1
C      DO 4 N = K,NMAX
C      B(N+1) = 0.0
C      IF(NMIN.EQ.0)B(1) = 1.0
C      RETURN
C
C      EPS = OK/10.
C      IF(EPS.LT.1.0E-14)EPS = 1.0E-14
C      CALL IBESS(X,NMAX,EPS,BIP,K)
C      B(NMAX+1) = BIP
C      IF(NMIN.EQ.NMAX)RETURN
C      N = NMAX - 1
C      CALL IBESS(X,N,EPS,BI,K)
C      B(N+1) = BI
C      IF(NMIN.EQ.N)RETURN
C      F = N
C      BIM = 2.0*F*BI/X + BIP
C      BIP = BI
C      BI = BIM
C      N = N - 1
C      GO TO 10
C      END

```

```

C
C      SUBROUTINE IBESS(X,N,EPS,BI,K)
C      PETER P. WINTERSTEINER, ARCON CORP., WALTHAM, MA.      8/2/79
C
C      SUBROUTINE IBESS, USED TO CALCULATE THE MODIFIED BESSEL FUNCTION
C      IN(X) FROM EITHER AN ASYMPTOTIC SERIES OR AN ASCENDING SERIES,
C      DEPENDING ON THE VALUES OF THE ARGUMENT, ORDER, AND ACCURACY DE-
C      Sired. THE ASCENDING SERIES IS EQUATION 9.6.10 ON P. 375 OF THE
C      NBS HANDBOOK; THE ASYMPTOTIC SERIES IS EQUATION 9.7.1 ON P. 377.
C
C      THE ASCENDING SERIES IS GENERALLY USED FOR SMALL ARGUMENTS. THE
C      ASYMPTOTIC SERIES FOR LARGE ARGUMENTS. SPECIFICALLY, THE ASCENDING
C      SERIES IS USED IF AND ONLY IF:
C
C      1) X < 5
C      OR 2) X < LEXP + 1      (N = 0 ONLY)
C      X < LEXP + ABS(N)
C
C      WHERE
C      X = ARGUMENT OF BESSEL FUNCTION (POSITIVE, REAL)
C      N = ORDER OF BESSEL FUNCTION (-100 < N < 100)
C      EPS = 10*(-LEXP) = DESIRED ACCURACY (LEXP.LE.14)
C
C      OTHERWISE THE ASYMPTOTIC SERIES IS FASTER. OTHER ELEMENTS IN THE

```



```

      AM = 1.0D+00
10  K = K + 1
      IS = -IS
      FK = K
      SIGN = IS
      FKN = MU - (2*K-1)**2
      AK = AK*FKN/(FK*8.0D+00*XD)
      SUM = SUM + SIGN*AK
      IF(K.LE.NN)GO TO 10
      FKN = NP - K
      E = DABS(AK*FKN/SUM)
      IF(E.LT.EPS)GO TO 20
      IF(NP.GT.K)GO TO 10
20  IF(E.EQ.0.0)K = K + 400
      FK = DSORT(2.0*PI*XD)
      BI = SUM/FK
      IF(MFF.EQ.0)BI = BI*EXP(X)
      RETURN
C
C
C   ASCENDING SERIES
C
2  NP = 0.5*(SORT(X*X + E*E) - E) + 2.5
   AK = 1.0D+00
   SUM = 1.0D+00
   X2 = XD*XD/4.
C
C   SUM POWER SERIES AT LEAST TO POINT OF DIMINISHING TERMS
C
DO 30 K = 1,NP
   FKN = K
   FKN = K + NN
   AK = AK*X2/(FK*FKN)
30  SUM = SUM + AK
C
C   SUM SERIES UNTIL FRACTIONAL ERROR IS LESS THAN EPS
C
K = NP
35  K = K + 1
   IF(K.GT.500)GO TO 40
   FK = K
   FKN = K + NN
   AK = AK*X2/(FK*FKN)
   SUM = SUM + AK
   E = AK*FK*FKN/((FK*FKN - X2)*SUM)
   IF(E.GE.EPS)GO TO 35
40  CONTINUE
   FK = 1.0D+00
   IF(NN.LE.0)GO TO 50
   DO 45 J = 1,NN
      FKN = J
45  FK = FK*FKN
50  CONTINUE
   BI = SUM*(XD/2.0)**NN/FK
   IF(MFF.EQ.1)BI = BI*EXP(-X)
   RETURN
   END
C
C   SUBROUTINE PERIM(PHI,RHO,DRDP)
C
C   EVALUATE RHO(PHI) ON THE PERIMETER.
C   EVALUATE D(RHO)/D(PHI) ON THE PERIMETER
C
REAL N

```

```

C      IER=0
C      COMMON/A,B,P2,P1,R,N,D,DK,NQM,NOT,NSM,NT2,EO
      IF(R.NE.1.0.OR.N.NE.1.0)GO TO 100
      RHO = 1.0
      DRDP = 0.0
      RETURN

C
C 100 CONTINUE
      CP = COS(PH1)
      SP = SIN(PH1)
      PN = 2.0*N
      IF(PH.LT.85.)GO TO 110
      DD = PH*ALOG(CP/R)/2.302585
      IF(DD.LT.-280.1CP = 0.0
      DD = PH*ALOG(SP)/2.302585
      IF(DD.LT.-280.1SP = 0.0
110 CONTINUE
      DD = (CP/R)*PN + SP*PN
      PN = 1.0/PN
      RHO = 1.0/(DD*PN)

      PN = 2.0*N - 2.0
      AA = SP*PN - ((CP/R)*PN)/(R*R)
      DRDP = -RHO*SP*CP*AA/DD
      RETURN
      END

      SUBROUTINE BESJ
      PURPOSE COMPUTE THE J BESSEL FUNCTION FOR A GIVEN ARGUMENT AND ORDER
      USAGE CALL BESJ(X,N,BJ,D,IER)
      DESCRIPTION OF PARAMETERS
      X -THE ARGUMENT OF THE J BESSEL FUNCTION DESIRED
      N -THE ORDER OF THE J BESSEL FUNCTION DESIRED
      BJ -THE RESULTANT J BESSEL FUNCTION
      D -REQUIRED ACCURACY
      IER-RESULTANT ERROR CODE WHERE
      IER=0 NO ERROR
      IER=1 N IS NEGATIVE
      IER=2 X IS NEGATIVE OR ZERO
      IER=3 REQUIRED ACCURACY NOT OBTAINED
      IER=4 RANGE OF N COMPARED TO X NOT CORRECT (SEE REMARKS)

      REMARKS
      N MUST BE GREATER THAN OR EQUAL TO ZERO, BUT IT MUST BE
      LESS THAN
      20*X-X**2/3 FOR X LESS THAN OR EQUAL TO 15
      90*X/2 FOR X GREATER THAN 15

      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
      NONE

      METHOD
      RECURRENCE RELATION TECHNIQUE DESCRIBED BY H. GOLDSTEIN AND
      R.M. THALER,"RECURRENCE TECHNIQUES FOR THE CALCULATION OF
      BESSEL FUNCTIONS",M.T.A.C.,V.13,PP.102-108 AND I.A. STEGUN
      AND M. ABRAMOWITZ,"GENERATION OF BESSEL FUNCTIONS ON HIGH
      SPEED COMPUTERS",M.T.A.C.,V.11,1957,PP.255-257

```

```

C      SUBROUTINE BESJ(X,N,BJ,D,IER)
      I = 0
      IF(N.EQ.0.AND.X.EQ.0.)BJ = 1.0
      IER = 0
      IF(X)30,200,10
      10 IF(N)20,31,31
      20 IER = 1
      RETURN
      30 IER=2
      RETURN
      31 IF(X-15.)32,32,34
      32 NTEST=20.+10.*X** 2/3
      GO TO 36
      34 NTEST=90.*X/2.
      36 IF(N-NTEST)40,38,38
      38 IER=4
      RETURN
      40 BPREV = 0.0

C      COMPUTE STARTING VALUE OF M
C
C      IF(X-5.)50,60,60
      50 MA=X+6.
      GO TO 70
      60 MA=1.4*X+60./X
      70 MB=N+FIX(X)/4+2
      MZERO=MAX(MA,MB)

C      SET UPPER LIMIT OF M
C
C      MMAX=NTEST
      100 DO 150 M=MZERO,MMAX,3
      C
      C      SET F(M).F(M-1)
      FM1=1.0E-28
      FM= 0
      ALPHA=.0

      IF(M-(M/2)*2)120,110,120
      110 JT=-1
      GO TO 130
      120 JT=1
      130 M2=M-2
      DO 160 K=1,M2
      MK=M-K
      BMK+2.=FLOAT(MK)*FM1/X-FM
      FM=FM1
      FM1=BMK
      IF(MK-N-1)150,140,150
      140 BJ=BMK
      150 JT=-JT
      S=1+JT
      160 ALPHA=ALPHA+BMK*S
      BMK+2.=FM1/X-FM
      IF(N)180,170,180
      170 BJ=BMK
      180 ALPHA=ALPHA+BMK
      BJ=BJ/ALPHA
      IF(ABS(BJ-BPREV)-ABS(D*BJ))200,200,190
      190 BPREV=BJ
      IER=3
      200 RETURN
      END
C      SUBROUTINE BESK

```



```

28 P=GO
C      JRN
C      COMPUTE K1 USING POLYNOMIAL APPROXIMATION
29 G1=A*(1.2533141+.4699927*T(1)-.1465583*T(2)+.1280427*T(3)
30 2-.1735432*T(4)+.2847618*T(5)-.4594342*T(6)+.6283381*T(7)
31 3-.6632295*T(8)+.5050239*T(9)-.2581304*T(10)+.07860001*T(11)
32 4-.01082418*T(12))*C
33 IF(N-1)20.30.31
34 BK=G1
35 RETURN
36 FROM KO,K1 COMPUTE KN USING RECURRENCE RELATION
37 RETURN
38 DO 35 J=2,N
39 GJ=2.*(FLOAT(J)-1)*G1/X*GO
40 IF(GJ-1.0E76)33.33.32
41 IER=4
42 GO TO 34
43 GO TO 34
44 GO=G1
45 G1=GJ
46 BK=GJ
47 RETURN
48 B=X/2
49 A=.5772157+ALOG(B)
50 C=B*B
51 IF(N-1)37.43.37
52 COMPUTE K0 USING SERIES EXPANSION
53 GO=-A
54 X2J=1.
55 FACT=1.
56 HJ=0
57 DO 40 J=1,6
58 RJ=1./FLOAT(J)
59 X2J=X2J*C
60 FACT=FACT*RJ*RJ
61 HJ=HJ+RJ
62 GO=GO+X2J*FACT*(HJ-A)
63 IF(N)43.42.43
64 BK=GO
65 RETURN
66 COMPUTE K1 USING SERIES EXPANSION
67 X2J=B
68 FACT=1.
69 HJ=1.
70 G1=1./X+X2J*(.5+A-HJ)
71 DO 50 J=2,8
72 X2J=X2J*C
73 RJ=1./FLOAT(J)
74 FACT=FACT*RJ*RJ
75 HJ=HJ+RJ
76 G1=G1+X2J*FACT*(.5*(A-HJ)+FLOAT(J))
77 IF(N-1)31.52.31
78 BK=G1
79 RETURN
80 END

```


Program Listing

- COAX -

(Reference: Vol. I, Section II.4)

```

PROGRAM COAX(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
PROGRAM COAX FINDS THE MODES OF A 3-MEDIUM DIELECTRIC WAVEGUIDE
OF ARBITRARY CROSS-SECTION. THE WEAKLY-GUIDING CONDITION IS
ASSUMED TO HOLD TRUE. THE INDICES OF REFRACTION OF THE THREE REGIONS
ARE SUCCESSIVELY LESS AS ONE PROGRESSES OUTWARD.

```

```

INTEGER S,SC,FLAG
REAL NL,NLL,N1,N2
DIMENSION PR(10),TABLE(51,3),INDEX(51)
DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24)
DIMENSION ROOT(20,14),ROOM(20,4),IROOT(20,3)
COMMON/A/B,P2,P1,NOT,NOT2,QQ
COMMON/B/STORE,INOX,GD,C,CON1,CON2
DATA (PR(1),1=1,3)/10H CIRCLE,10H ELLIPSE,10H SQUARE /
DATA (PR(1),1=4,7)/10H S' ELLIPSE,10H RECTANGLE,6M SINE /
DATA (PR(1),1=8,9)/10H PLIPIDED,10H CUSPED /

```

LIST OF VARIABLES WITH SINGLE ASSIGNED PURPOSES

```

ACC ACCURACY TO WITHIN WHICH THE ROOT-LOCATION PROCEDURE FINDS THE ROOTS
B NORMALIZED FREQUENCY (READ IN)
C VECTOR OF COEFFICIENTS IN THE FIELD EXPANSION (DESIRED RESULT)
CON1 FACTOR USED TO KEEP DETERMINANT FROM UNDERFLOWING WHEN QQ >> 1
CON2 FACTOR USED TO KEEP DET FROM OVERFLOWING
DET VALUE OF THE DETERMINANT OF THE MATRIX GD
DRDP D(RHO)/D(PHI) ON THE PERIMETER OF THE GUIDE
FLAG OVERFLOW INDICATOR (SEE SUBROUTINE CHECK) (INTEGER)
GD MATRIX OF COEFFICIENTS
ICON EXPONENT USED TO PRESET CON2 TO 10**(-ICON) (READ IN)
ID # OF FUNCTION EVALUATIONS DURING ROOT-LOCATION PROCEDURE
INDIAG DIAGNOSIS CODE (SEE COMMENTS) (READ IN)
INDEX ARRAY WITH TABLE INDICES NEAR WHICH ROOTS OCCUR; ALSO NOTES OVERFLOW
INX ARRAY FOR STORING VALUES OF INDICES USED IN FIELD EXPANSION
IROOT ARRAY FOR STORING INFORMATION USED IN PRINTING RESULTS AT THE END
JDEXT COUNTER TO KEEP TRACK OF THE NUMBER OF SEPARATE CASES COMPLETED
L INDEX OF FIRST TERM IN FIELD EXPANSION (READ IN)
LLS VALUE OF L FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
NL,N2 SUPERELLIPSE INDICES (REAL) (READ IN)
NLL VALUE OF N1 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
NOP # OF INTEGRATION POINTS (READ IN)
NOT # OF TERMS IN EACH SUBMATRIX (READ IN)
NOT2 DIMENSION OF DETERMINANT = 2*NOT
NP2 # OF VALUES OF P2 TAKEN TO CONSTRUCT TABLE (READ IN)
NR # OF ROOTS FOUND
PHI ANGLE IN (RHO, PHI) COORDINATE SYSTEM
PI
P1 LARGEST VALUE OF P2 USED IN TABLE (READ IN)
P2 SMALLEST VALUE OF P2 USED IN TABLE (READ IN)
PMIN ARRAY FOR STORING ALPHANUMERIC PROGRAM INFORMATION
PR PROPAGATION CONSTANT (DESIRED RESULT)
QQ RATIO OF NUMERICAL APERTURES (1-3 TO 1-2) (READ IN)
RHO ASPECT RATIOS (READ IN)
R1,R2 RADIUS IN (RHO, PHI) COORDINATE SYSTEM
RL VALUE OF R1 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
RL VALUE OF R2 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
ROOM ARRAY FOR STORING OUTPUT INFORMATION
ROOT ARRAY FOR STORING OUTPUT INFORMATION, INCLUDING VALUES OF P2
S INDEX OF LAST TERM IN FIELD EXPANSION (INTEGER)

```

```

C SC INDICATES SINE OR COSINE EXPANSION (INTEGER) (READ IN)
C SW ARRAY FOR STORING FUNCTIONS USED REPEATEDLY, WHICH DND ONLY ON PHI
C SZ RATIO OF OUTER SEMIMINOR AXIS TO INNER (SZ.GT.1)
C SZL VALUE OF SZ FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
C TO TABLE INITIAL TABLE OF VALUES OF DETERMINANT FOR DIFFERENT VALUES OF P2
C T0 INITIAL CP TIME
C T1 LATER CP TIME
C .....
100 FORMAT(6I5)
101 FORMAT(////,16X,'PROGRAM COAX-//,6X,DATE:',A10,
1. TIME:',A10)
102 FORMAT(////,1X,12,'--TERM TRUNCATION FOR ALL CASES: NOT2 =,13)
103 FORMAT(' IDIAG = DIAGNOSIS CODE, FOR ALL CASES =,13)
104 FORMAT(' ACC = ACCURACY OF ROOT LOCATION = 1.0E-,12)
105 FORMAT(' NP2 = NUMBER OF INITIAL TABLE POINTS =,13)
106 FORMAT(//,10X,N,6X,'P2-',8X,'DET',/)
107 FORMAT(//,F10.6,E14.7)
108 FORMAT(' NOP = NUMBER OF INTEGRATION POINTS =,13)
109 FORMAT(//,18X,'CASE #,12,/,18X,'L =,14,/,18X,'S =,14,/,
116X,'N1 =,F8.4,/,16X,'R1 =,F8.4,/,
216X,'N2 =,F8.4,/,16X,'R2 =,F8.4,/,16X,'SZ =,F8.4,/,
316X,'B =,F8.4,/,16X,'QO =,F8.4,/)
110 FORMAT(5X,'PMAX =,F6.4,10X,'PMIN =,F6.4,/)
111 FORMAT(//)
112 FORMAT(//,1X,16,' EXPANSION: INNER CROSS-SECTION IS A,A10,/,
119X,'OUTER CROSS-SECTION IS A,A10)
113 FORMAT(' $$$$$$ ROOT IS EXACT')
114 FORMAT(1X)
115 FORMAT(5X,'EXEC TIME FOR TABLE =,F7.3,' SEC')
116 FORMAT(//,,' SUMMARY OF COAX RESULTS:',/)
15X,'SHAPE',19X,N,7X,'R',6X,'SZ/QO',5X,'B',6X,'ROOTS (P2)...')
117 FORMAT(//,13,1X,R9,12,13,2X,16,F8.4,F7.4,F8.3,F7.3,9F9.5)
118 FORMAT(1X,A3,' =,9E14.5,/,6X,9E14.5,/,7X,9E14.5)
121 FORMAT(215,9F5.2,15)
122 FORMAT(//,,' COEFFS ARE =,6(A1,A(,12,) =,E11.4,.,),/,12X,
16(A1,A(,12,) =,E11.4,.,))
123 FORMAT(' ////////////////////////////////////////')
124 FORMAT(16,' ROOT(S) IDENTIFIED FOR CASE #,13,/)
125 FORMAT(' CASE #,13,' ROOT #,12,' P2 =,F10.8,' +/-,E9.3,.,)
113,' NEW FN EVALS: DET =,E13.5)
126 FORMAT(//,,' EXEC TIME THIS CASE =,F7.3,' SEC',/)
127 FORMAT(//,13,1X,R9,12,13,2X,16,F8.4,F7.4,F8.3,F9.5,7F10.8,/,58X,
1F10.8)
128 FORMAT(//,,' $$$$$$ MORE THAN 8 ROOTS, LAST=,13,
1,' NOT LISTED IN SUMMARY',/)
129 FORMAT(4X,R9,13X,F8.4,F7.4,F8.3,/)
130 FORMAT(1X)
131 FORMAT(//,,' $$$$$$ ERROR---NOP = 1 BUT CROSS-SECTION IS NOT A ',
1,'CIRCLE $$$$$$')
132 FORMAT(12X,6(A1,A(,12,) =,E11.4,.,),/)
133 FORMAT(//,,' $$$$$$ CASE #,12,' NOTE ADJUSTED VALUE OF SZ =,
1F6.3,' $$$$$$')
134 FORMAT(' $$$$$$ CASE #,12,' SZ ADJUSTED TO 1.01, UNDERFLOW ',
1,'LIKELY $$$$$$')
135 FORMAT(//,,' CONT =,E9.3,/)
136 FORMAT(//,40X,'$$$$$ IF JOB BOMBS AT 1ST OR 2ND TABLE ENTRY DUE',
1,' TO OVERFLOW IN MATIN=,/,40X,'$$$$$ (CHECK ARRAY DDD IN ',
2,' COMMON/M/ WITH A PMD), TRY OVER AGAIN WITH=,/,40X,
3,'$$$$$ ICON >=,12,' TO PRESET CON2 TO A VALUE SMALLER THAN ',
4,'UNITY: 1.0E-ICON')
137 FORMAT(' CON2 IS INITIALLY 1.0E-,12,/)
C .....
C .....
C .....

```

C INITIALIZE CONSTANTS FOR ENTIRE JOB

```

( 3.1415926535898
2 CALL DATE(PN)
  CALL TIME(W)
  RL = 0.0
  NL = 0.0
  RLL = 0.0
  NLL = 0.0
  LLL = -100
  LLS = -100
  SZL = -100.

```

```

NOT = NUMBER OF TERMS TAKEN BEFORE TRUNCATION
ACCURACY OF HALF-INTERVAL SEARCH IS 10**(-5)
NP2 = NUMBER OF INTERVALS IN THE INITIAL TABLE
NOP = NUMBER OF INTEGRATION POINTS IN THE RANGE (0,PI/2)
IDIAG = DIAGNOSIS CODE: 0 = NORMAL CONDITION (DEFAULT)
                        1 = INITIAL TABLE, FINAL MATRIX PRINTED
                        2 = SUMMARY PRINTED AT END (VALUES OF P2)
                        3 = DETAILS OF ROOT-LOCATION PRINTED
                        4 = UNDERFLOW, OVERFLOW CHECKS PERFORMED
DEF: 1.0E-3
DEF: 10
DEF: 50

```

NOTE: IDIAG = 4 SHOULD BE USED ONLY FOR LARGE VALUES OF NOT WHEN UNDERFLOW OR OVERFLOW IS ANTICIPATED, SINCE IT CAUSES COMPUTATIONS TO TAKE PLACE WHICH ARE UNNECESSARY IN NORMAL CIRCUMSTANCES.

- IF THE CROSS-SECTION IS CIRCULAR, THE PHI-INTEGRALS CAN BE DONE IN CLOSED FORM. TO SAVE TIME, ONE CAN SET NOP = 1, LEAVING ONLY ONE "INTEGRATION POINT". IN THIS CASE ADJUSTMENTS IN THE VALUES OF THE TRIG FUNCTIONS ARE MADE AT STATEMENT 87 SO THAT THE INTEGRALS WILL COME OUT RIGHT. NOTE THAT IF NOP = 1, NOT MUST ALSO EQUAL 1 OR ELSE THE OFF-DIAGONAL MATRIX ELEMENTS MAY BE NONZERO AND THEREFORE INCORRECT.

READ IN INFORMATION WHICH REMAINS FIXED FOR MANY CASES.
RETURN TO STATEMENT 2 ONLY AFTER DUMPING RESULTS FOR THESE CASES.
A BLANK CARD HERE TERMINATES PROGRAM EXECUTION.

```

READ(5,100)NOT,S,NP2,NOP,IDIAG
IF(NOT.EQ.0)STOP
WRITE(6,130)
WRITE(6,123)
WRITE(6,101)PN,W
IF(NOP.GT.12)NOT = 12
IF(S.LE.0)S = 3
IF(S.GT.12)S = 12
ACC = 10.0**(-5)
IF(NP2.LT.1)NP2 = 10
IF(NP2.GT.50)NP2 = 50
IF(NOP.GT.200)NOP = 200
IF(NOP.EQ.0)NOP = 50
IF(NOP.EQ.1)OT = 1
NOT2 = 2*NOT
M = -1
IF(IDIAG.GT.3)CALL CHECK(DEF,NOT2,M,NP2,PMIN,1,FLAG,CON2)

```

PRINT THE FIXED INFORMATION READ FROM THE FIRST DATA CARD

```

WRITE(6,102)NOT,NOT2
WRITE(6,104)S
WRITE(6,105)NP2
WRITE(6,108)NOP
WRITE(6,103)IDIAG

```



```

C THE K BESSEL FUNCTIONS, INDIVIDUALLY, MAY BE SMALL WITHOUT UNDERFLOWING,
C THEIR PRODUCTS (IN NLS, THE 2ND QUADRANT OF GD) MAY UNDERFLOW, OR
C (WHOLE DETERMINANT MAY UNDERFLOW, THEREFORE SET THE FIRST, SECOND, OR
C TO A LARGE VALUE AND MULTIPLY ALL BESSEL FUNCTIONS USED IN THE FIRST
C AND SECOND QUADRANTS BY THIS CONSTANT TO KEEP THE DETERMINANT WITHIN
C BOUNDS. IN OTHER WORDS, MULTIPLY THE FIRST NOT HOMOGENEOUS EQUATIONS
C BY CON1**2.
C
C2 CON1 = 1.0
W = SZ*W
IF((NOT*W).LT.400.)GO TO 64
CON1 = EXP(W/2.0)
CON1 = ALOG10(CON1)
I = CON1
CON1 = 10.0**I
C
C
C
C DETERMINE S, THE INDEX OF THE LAST TERM IN THE FIELD EXPANSION
C
C4 S = L + 2*(NOT-1)
IF(R1.NE.1.0.OR.R2.NE.1.0)GO TO 65
I = L/2
I = 2*I
IF(I.NE.L)GO TO 65
S = L + 4*(NOT-1)
C
C
C
C PRINT INPUT DATA
C
C5 WRITE(6,111)
WRITE(6,123)
WRITE(6,111)
WRITE(6,109)DEX,L,S,N1,R1,N2,R2,SZ,B,QQ
IF(PMAX.NE.1.0.OR.PMIN.NE.0.0)WRITE(6,110)PMAX,PMIN
IF(IDIAG.GT.0.AND.CON1.NE.1.)WRITE(6,135)CON1
IF(IDIAG.GT.3.AND.ICON.NE.0)WRITE(6,137)ICON
C
C
C INITIALIZE STORAGE ARRAYS FOR THIS CASE
C
C6 DO 70 I = 7,14
ROOT(JDEX,1) = 0.0
ROOT(JDEX,3) = N1
ROOT(JDEX,4) = R1
ROOT(JDEX,5) = SZ
ROOT(JDEX,6) = B
ROOM(JDEX,2) = N2
ROOM(JDEX,3) = R2
ROOM(JDEX,4) = QQ
ROOT(JDEX,1) = 0
ROOT(JDEX,2) = L
ROOT(JDEX,3) = S
C
C
C IDENTIFY SHAPES OF CROSS-SECTIONS
C
C7 J = SC + 5
I = 4
IF (N1.GT.1.0) GO TO 90
IF (N1-.5) 94,95,96
C8 I = 9
GO TO 92

```

```

98 I = 8
99 ( TO 92
96 ( (N1-NE.1.0) GO TO 92
1=2
IF (R1.EQ.1.0) I=1
90 IF (N1.LT.30.) GO TO 92
I=5
IF (R1.EQ.1.0) I=3
92 CONTINUE
W = PR(I)
IF (NOP.EQ.1.AND.I.NE.1) WRITE(6.131)
I=4
IF (N2.GT.1.0) GO TO 50
IF (N2-.5) 54.55.56
94 I=9
GO TO 52
95 I=8
GO TO 52
96 IF (N2.NE.1.0) GO TO 52
I=2
IF (R2.EQ.1.0) I=1
50 IF (N2.LT.30.) GO TO 52
I=5
IF (R2.EQ.1.0) I=3
92 CONTINUE
WRITE(6.112) PR(J), W, PR(I)
IF (NOP.EQ.1.AND.I.NE.1) WRITE(6.131)
ROOT(JDEX,2) = PR(J)
ROOT(JDEX,1) = W
ROOM(JDEX,1) = PR(I)
.....
C
C
C
PN = NOP
W = PI/(PN*2.0)
IF (RL.NE.R1.OR.RLL.NE.R2) GO TO 71
IF (NL.NE.N1.OR.NLL.NE.N2) GO TO 71
IF (SZ.NE.SZL) GO TO 71
GO TO 79
71 CONTINUE
RLL = R2
NLL = N2
RL = R1
NL = N1
SZL = SZ
C
C
C
CALCULATE AND STORE RHO(PHI) AND D(RHO)/D(PHI) FOR EACH BOUNDARY
DO 75 I = 1,NOP
PH = 2*(NOP-I) + 1
PHI = W*PN/2.0
CALL PERIM(N1,R1,PHI,RHO,DRDP)
STORE(1,1) = RHO
STORE(1,2) = DRDP
CALL PERIM(N2,R2,PHI,RHO,DRDP)
STORE(1,3) = RHO*SZ
STORE(1,4) = DRDP*SZ
75 STORE(1,4) = DRDP*SZ
C
C
C
CALCULATE AND STORE COS(M*PHI) AND SIN(M*PHI) WHERE M CAN BE L,.....,S.
VALUES OF M ARE STORED AS INDX.
79 CONTINUE
IF (L.EQ.LLL.AND.S.EQ.LLS) GO TO 89
LLL = L
LLS = S
INDX(1) = L

```


[illegible]

```

C C COEFFICIENTS. THE FINAL COEFFICIENTS ARE THE COEFFICIENTS OF THE 1 AND K
C C ! SEL FUNCTIONS IN REGION 2. TO GET THE COEFFICIENTS OF J AND M
C C ! SEL FUNCTIONS, IT IS NECESSARY TO MULTIPLY BY COMPLEX ( J AND M
C C ! TORS.

DO 255 K = 1,NOT
J = INDX(K)
PN = (-1.0)**J
C(K) = PN*C(K)
255 C(K+NOT) = 2.0*PN*C(K+NOT)/PI
PN = 1H
WRITE(6,122)((PN,INDX(K),C(K)),M=1,NOT)
J = NOT + 1
WRITE(6,132)((PM,INDX(K+NOT),C(K)),K=J,NOT2)
WRITE(6,114)
IF(DET.EQ.0.0)WRITE(6,113)
250 CONTINUE
C
CALL SECOND(T1)
T1 = T1 - TO
WRITE(6,126)T1
GO TO 1
C
C .....
C AFTER ALL CASES (20 OR LESS) HAVE BEEN COMPLETED, PRINT A TABLE
C OF RESULTS GIVING THE ROOTS FOR EACH CASE.
C .....
999 WRITE(6,123)
WRITE(6,123)
WRITE(6,123)
WRITE(6,116)
DO 900 I = 1,JDEX
K = IROOT(I,1) + 6
IF(1DIAG.LE.1)WRITE(6,117)I,ROOT(I,1),(IROOT(I,M),M=2,3),
1(ROOT(I,J),J=2,K)
IF(1DIAG.GE.2)WRITE(6,127)I,ROOT(I,1),(IROOT(I,M),M=2,3),
1(ROOT(I,J),J=2,K)
900 WRITE(6,129)(ROOM(I,M),M=1,4)
GO TO 2
END
SUBROUTINE CHECK(DET,NOT2,M,MP2,PMIN,UF,FLAG,CON2)
SUBROUTINE CHECK TABLE ENTRIES TO DETERMINE IF:
1) A ZERO ENTRY IS AN ACTUAL ROOT OR AN UNDERFLOW ERROR
2) OVERFLOW IS LIKELY TO OCCUR IN THE NEXT TABLE ENTRY
IN THE LATTER CASE, IT REDEFINES (IF POSSIBLE) THE FACTOR CON2 IN AN
ATTEMPT TO SCALE THE DETERMINANT (BY AN AMOUNT CON2**NOT2) DOWN TO
SMALLER VALUES AND THUS PREVENT OVERFLOW. OVERFLOW ACTUALLY OCCURS IN
INTERMEDIATE STEPS OF THE DETERMINANT DETERMINATION.
C
C NOTE THAT THE OVERFLOW-AVOIDANCE PROCEDURE IS NOT FOOLPROOF: THE ANTI-
C CIPATION OF OVERFLOW RESULTS WHEN A POLYNOMIAL FIT TO POINTS COMPRISING
C THE LOGARITHMS OF THE LARGEST INTERMEDIATE VALUES OF DETER IN SUBROU-
C TIME MATIN FOR PREVIOUS TABLE ENTRIES, IS EXTRAPOLATED TO THE NEXT EN-
C TRY AND EXCEEDS 320 AT THAT POINT. HOWEVER THE APPROXIMATION AFFORDED
C BY THE POLYNOMIAL CAN IN SOME CIRCUMSTANCES BE AN UNDERESTIMATE OF THE
C ACTUAL NEXT ENTRY AND THEREFORE FAIL TO ANTICIPATE AN ACTUAL OVERFLOW
C SITUATION.
C
C .....
C INTEGER UF,FLAG
C C .....

```

```

      DIMENSION DDD(24),PVT(24),F(4)
      NON/M/DDD,PVT

C
100 FORMAT(40X,'$$$$$ THIS TABLE ENTRY UNDERFLOWS. SIGN OF DET',
11. WOULD BE ',A4,'TIVE $$$$$$')
101 FORMAT(40X,'$$$$$ ANTICIPATE OVERFLOW ON NEXT TABLE ENTRY. RE',
11-SET CON2 = 1.00E-14/,40X,'$$$$$ TOTAL FACTOR IS NOW ',
21.00E-14)
102 FORMAT(40X,'$$$$$ 21X,*,FO,...,FN **,4(FB.1.,*),F6.1)
103 FORMAT(40X,'$$$$$ ANTICIPATE UNAVOIDABLE OVERFLOW ON NEXT TABLE',
11-ENTRY. EXIT LOOP.*/40X,'$$$$$ RESET LST TO*,13.,*)
104 FORMAT(40X,'$$$$$ THIS PROBABLY WON'T PROVIDE A SUFFICIENT',
11-CUSHION WITH FJ **,F6.1)

C
IF(M.LT.0)GO TO 520
DTT = ABS(DET)
UF = 0

C
CHECK UNDERFLOW POSSIBILITY. WE ASSUME THAT NONE OF THE PIVOT ELEMENTS
(ARRAY PVT) USED IN MATIN UNDERFLOW, BUT THAT THE PRODUCT OF ALL PIVOT
ELEMENTS (THAT IS, THE DETERMINANT) MAY UNDERFLOW. (THE RUNNING PRODUCT
IS STORED IN ARRAY DDD.) THEREFORE PIVOT ELEMENTS 1 TO NOT2 ARE ALL DE-
FINED AND NONZERO WHEN THERE IS AN UNDERFLOW PROBLEM. WHEN THE DETERMI-
NANT OF THE MATRIX IS TRULY ZERO, THE LAST PIVOT ELEMENT IS NOT DEFINED-
SINCE THE EXIT TO STATEMENT 750 FROM STATEMENT 105 (IN MATIN) BYPASSES
THE DEFINITION OF PIVOT. (HERE, 'UNDEFINED' MEANS HAVING A VALUE EQUAL
TO RST.)

C
IF UNDERFLOW OCCURS, SET UF = 1 AND CHECK TO SEE WHAT SIGN THE DETERMI-
NANT WOULD HAVE HAD. MAKE ADJUSTMENTS IN MAIN.

C
IF(DTT.NE.0.0)GO TO 300
IF(PVT(NOT2).EQ.RST)GO TO 300
UF = 1
DO 200 I = 1,NOT2
IF(DDD(I).NE.0.0)GO TO 200
DDD(I) = DDD(I-1)*(PVT(I)/ABS(PVT(I)))
200 CONTINUE
CD = 4*POST
IF(DDD(NOT2).LT.0.0)CD = 4*MEGA
WRITE(6,100)CD

C
CHECK OVERFLOW POSSIBILITIES. OVERFLOW CAN OCCUR IN AN INTERMEDIATE STEP
OF MATIN. SO WE SEARCH FOR THE LARGEST VALUE THAT DDD TAKES ON FOR EACH
TABLE ENTRY (DMAX) AND TRY TO ANTICIPATE IF, FOR THE NEXT TABLE ENTRY,
THIS VALUE WILL BECOME TOO LARGE TO HANDLE. IF OVERFLOW SEEMS LIKELY,
CON2 IS DETERMINED TO BE THE SMALLEST POSITIVE NUMBER OF THE FORM 10**K
(K INTEGER) WHICH STILL WILL BE LARGE ENOUGH TO PREVENT UNDERFLOW WHERE-
EVER (INCLUDING SUBROUTINE RTT) IT MIGHT OCCUR.

C
300 IF(M.GT.NP2)GO TO 500
DMAX = -RST
DO 320 I = 1,NOT2
CD = ABS(DDD(I))
IF(CD.LT.DMAX)GO TO 320
DMAX = I
DMAX = CD
320 CONTINUE
FF = ALOG10(DMAX)
IF(M.GT.1)GO TO 330
FLAG = 0
DO 325 I = 1,4
325 F(I) = FF
FN = FF
FAC = 1.0
IF(PMIN.EQ.0..AND.M.EQ.NP2)FAC = 3.0

```

```

330 L = M
      ( L.GT.5) L = 5
      TO(360,340,350,370,380),L
C
C FORWARD-DIFFERENCE FORMULAE FOR EXTRAPOLATION TO NEXT TABLE ENTRY
C
340 FN = 2.0*FAC*FF - F(1)
      GO TO 380
350 FN = 3.0*FAC*FF - 3.0*F(2) + F(1)
      GO TO 380
360 L = 4
      DO 385 I = 1,3
365 F(I) = F(I+1)
370 FN = 4.0*FAC*FF - 6.0*F(3) + 4.0*F(2) - F(1)
380 F(L) = FF
      IF(FN.LT.320.)GO TO 500
C
C DEAL WITH OVERFLOW ANTICIPATED IN THE NEXT TABLE ENTRY BY REDEFINING
C CON2. CHECK CON2 TO SEE IF THIS HELPS. IF NOT, GIVE UP.
C
      IF(DDT.EQ.0)DDT = 1.0E-293
      CD = ALOG10(DDT)
      K = CD
      IF(K.GT.12)K = 12
      K = (280 + K)/NOT2
      IF(K.LE.0)GO TO 410
      CON2 = CON2*10.0**(-K)
      FLAG = 1
      CD = ALOG10(CON2) - .0001
      J = CD
      I = J*NOT2
      WRITE(6,101),J,I
      WRITE(6,102)(F(J),J=1,L),FN
      FN = FN - J*MAX*K
      IF(FN.GT.320.)GO TO 400
      DO 390 J = 1,L
390 F(J) = F(J) - J*MAX*K
      GO TO 500
C
C CAN'T DO MUCH ABOUT ANTICIPATED OVERFLOW. RESET LST IN MAIN.
C
400 WRITE(6,104)FN
410 FLAG = 2
      WRITE(6,103)M
      IF(K.LE.0)WRITE(6,102)(F(J),J=1,L),FN.
C
C RESET ARRAYS ODD AND PVT
C
500 DO 510 I = 1,NOT2
510 PVT(I) = RST
      RETURN
C
520 RST = (1./3.)*1.0E+321
      GO TO 500
      END
      SUBROUTINE RTTT(TABLE,ACC,PN,NOP,SC,M,ID,IDIAG)
C
C SUBROUTINE RTTT PRECISELY LOCATES THE ROOT SPECIFIED BY M (USING
C ELEMENTS M AND M+1 IN THE ARRAYS TABLE(M,1) AND TABLE(M,2)). AND THEN
C DETERMINES THE FIELD EXPANSION COEFFICIENTS C(J) CORRESPONDING TO
C THAT ROOT. ACC IS THE DESIRED ACCURACY OF THE ROOT. AND PN IS THE
C DIFFERENCE IN THE ACTUAL BOUNDS OBTAINED. ID IS THE NUMBER OF FUN-
C CTION EVALUATIONS NECESSARY. SC, IDIAG, AND NOP ARE AS IN MAIN. GO, THE
C FINAL MATRIX AT THE ROOT P2, IS ALSO RETURNED. (C, GO, AND P2 ARE ALL

```

```

C      IN COMMON BLOCKS.)
C      (
C      IER SC
C      DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24),SINV(24,24)
C      DIMENSION TABLE(S1,3)
C      COMMON/A/B,P2,P1,NOT,NOT2,GD
C      COMMON/B/STORE,INDX,GD,C,CON1,CON2
C
C 700 FORMAT(/, 'SEARCH ID=7X, X1=,11X, X2=,10X, F1=,9X, F2=,11X, DX=,
19X, P2=,8X, F=')
C 701 FORMAT(/, ' $$$$ ERROR IN HINTS OR REG, IER =,13, $$$$')
C
C      IF(IDIAG.GE.3)WRITE(6,700)
C      CON2 = TABLE(M+1,3)
C      K = 0
C      ID = 0
C      IF(M.LT.1000)GO TO 230
C      P2 = TABLE(M-1000,1)
C      CON2 = TABLE(M,3)
C      CALL CHEBY(NOP,SC)
C      ID = 1
C      PN = 0.0
C      GO TO 248
C
C 230 G1 = TABLE(M+1,2)
C      G2 = TABLE(M,2)
C      X1 = TABLE(M+1,1)
C      X2 = TABLE(M,1)
C      DN = TABLE(M,3)
C      IF(DN.NE.CON2)G2 = G2*((CON2/DN)**NOT2)
C      DN = X2 - X1
C
C      IF G1 AND G2 ARE TWO OR MORE ORDERS OF MAGNITUDE DIFFERENT, USE
C      A HALF-INTERVAL SEARCH TO NARROW THE RANGE X1 TO X2
C
C 235 NW = -G1/G2
C      IF(DN.LT.100..AND.NW.GT..01)GO TO 240
C      IF(DN.LT..ACC)GO TO 240
C      DN = DN/3.0
C
C      CALL HINTS(X1,X2,G1,G2,DN,PN,NOP,SC,ID,IDIAG,IER)
C      IF(IER.GT.0)WRITE(6,701)IER
C
C      K = K + ID
C      IF(PN.EQ.0.0)GO TO 248
C      GO TO 235
C
C      MODIFIED REGULA-FALSI METHOD TO GET THE ROOT PRECISELY
C
C 240 CALL REG(X1,X2,G1,G2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C      IF(IER.GT.0)WRITE(6,701)IER
C
C 248 ID = ID + K
C
C      CALCULATE THE COEFFICIENTS. SUBROUTINE LOAD COPIES (NOT2-1) OF THE
C      NOT2 ROWS OF GD INTO ARRAY SINV AND VECTOR C. SUBROUTINE MATIN SOLVES
C      THE (NOT2-1)X(NOT2-1) SYSTEM FOR THE COEFFICIENTS AND RETURNS THEM IN C.
C
C      CALL LOAD(GD,SINV,C,NOT2,M)
C      LST = NOT2 - 1
C      CALL MATIN(SINV,LST,C,DN,1)
C
C      STACK THE COEFFICIENTS IN THE PROPER ORDER IN VECTOR C, INCLUDING 1.000
C
C      IF(M.EQ.NOT2)GO TO 280
C      DO 270 J = M,LST

```

```

      K = NOT2 + M - J
270 ( ) = C(K-1)
280 (,4) = 1.0
      RETURN
      END
      SUBROUTINE LOAD(SS,SINV,C,NOT,1D)
      C
      C SUBROUTINE LOAD LOADS SS (NEE GD) INTO ARRAY SINV, ELIMINATING
      C THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
      C IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
      C LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
      C INATED. THUS, SINV IS (NOT-1) BY (NOT-1) IN DIMENSION, AND C IS A VEC-
      C TOR (NOT-1) IN LENGTH.
      C
      C REMEMBER, NOT IS WHAT IS CALLED NOT2 IN COAX. ALSO, ONLY THE FIRST
      C FOUR ELEMENTS OF EACH HALF OF THE DIAGONAL ARE SEARCHED FOR THE
      C SMALLEST VALUE.
      C
      C DIMENSION SS(24,24),SINV(24,24),C(24)
      C G = 1.0E+321
      C 1D = 1
      C K = NOT/2
      C IF(K.GT.4)K = 4
      C DO 200 J = 1,K
      C H = ABS(SS(J,J))
      C IF(H.GT.G)GO TO 200
      C G = H
      C 1D = J
      C 200 CONTINUE
      C
      C JJ = NOT/2 + 1
      C K = JJ + 3
      C IF(K.GT.NOT)K = NOT
      C DO 210 J = JJ,K
      C H = ABS(SS(J,J))
      C IF(H.GT.G)GO TO 210
      C G = H
      C 1D = J
      C 210 CONTINUE
      C
      C JJ = 0
      C DO 300 J = 1,NOT
      C IF(J.EQ.1D)GO TO 300
      C JJ = JJ + 1
      C C(JJ) = -SS(J,1D)
      C KK = 0
      C DO 299 K = 1,NOT
      C IF(K.EQ.1D)GO TO 299
      C KK = KK + 1
      C SINV(JJ,KK) = SS(J,K)
      C 299 CONTINUE
      C 300 CONTINUE
      C RETURN
      C END
      C SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,1D,1DIAG,1ER)
      C
      C SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE A ROOT
      C OF THE EQUATION DET(RIS) = 0 TO WITHIN A DESIRED ACCURACY ACC.
      C 1D IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
      C
      C INTEGER SC
      C DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24)
      C COMMON/A/B,P2,P1,NOT,NOT2,QQ
      C COMMON/B/STORE,INDX,GD,C,CON1,CON2
      C
      C 1D = 0

```

115

```

180 X1 = P2
   ( .G
      ( (FO-G).GT.0.0)F2 = F2/2.0
      GO TO 100
C
C IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE
C BEEN ALTERED IN MATIN) IS RECOMPUTED.
C
190 PN = 0.0
   CALL CHEBY(NOP,SC)
   ID = ID + 1
   RETURN
C
C ERROR RETURN: TOO MANY ITERATIONS
C
230 IER = 5
   RETURN
   END
C
C SUBROUTINE CHEBY(NOP,SC)
C
C INTEGER S,SC
C REAL MLS,NLS
C DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24)
C DIMENSION GDM(12,12),GDM1(12,12),GDP(12,12),GDO(12,12)
C DIMENSION BIST1(49),BIST2(49),BJST(49),BKST1(49),BKST2(49)
C DIMENSION BKST3(49)
C COMMON/A,B,P2,P1,NOT,NOT2,QQ
C COMMON/R/STORE,INDX,GD,C,CON1,CON2
C
C CHEBYSHEV INTEGRATION OF THE EXTERNAL FUNCTION G(PHI)
C
NMAX = INDX(NOT) + 1
IF(NMAX.EQ.1)NMAX = 2
DR = 0.0
NN = 0
PTS = NOP
M = PI/(PTS+2.0)
P = SORT(P2)
OMP = SORT(1.0 - P2)
DO 50 I = 1,NOT
DO 50 J = 1,NOT
GDM(I,J) = 0.0
GDM1(I,J) = 0.0
GDP(I,J) = 0.0
GDO(I,J) = 0.0
50 GDO(I,J) = 0.0
C
C LOOP DO 100 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
C
DO 100 JJJ = 1,NOP
OBTAIN RHO(PHI), COS(PHI), AND SIN(PHI)
RHI = STORE(JJJ,1)
RH2 = STORE(JJJ,3)
DR1 = STORE(JJJ,2)
DR2 = STORE(JJJ,4)
C
C AK1R1=K1*RHO1 FOR SURFACE 1-2
C AK2R1=K2*RHO1 FOR SURFACE 1-2
C AK3R2=K3*RHO2 FOR SURFACE 2-3
C AK2R2=K2*RHO2 FOR SURFACE 2-3
C
C AK1R1 = PI*B*RHI*OMP
C AK2R1 = PI*B*RHI*P
C AK2R2 = PI*B*RH2*P

```



```

C      PI=8*RM2*SQRT(00 - 1.0 + P2)
C      OBTAIN THE BESSEL FUNCTIONS WHICH WILL BE REQUIRED
C
C      IS(AK2R1) IS STORED IN VECTOR B1ST1:  B1ST1(1) = 10, B1ST1(2) = 11, ETC.
C      IS(AK2R2) IS STORED IN VECTOR B1ST2
C      JL(AK1R1) IS STORED IN VECTOR BJST
C      KS(AK2R1) IS STORED IN VECTOR BKST1
C      KS(AK2R2) IS STORED IN VECTOR BKST2
C      KL(AK3R2) IS STORED IN VECTOR BKST3
C
C      CALL BES1(AK2R1,0R,1,NMAX,B1ST1,NZ)
C      CALL BES1(AK2R2,0R,1,NMAX,B1ST2,NZ)
C      CALL BESJ(AK1R1,0R,NMAX,BJST,NZ)
C      CALL BESK(AK2R1,NH,1,NMAX,BKST1,NZ)
C      CALL BESK(AK2R2,NH,1,NMAX,BKST2,NZ)
C      CALL BESK(AK3R2,NH,1,NMAX,BKST3,NZ)
C
C      LOOP DO 100 I... CORRESPONDS TO L
C      LOOP DO 100 J... CORRESPONDS TO S
C
C      DO 100 I = 1,NOT
C      L = INDX(I)
C      XL = L
C
C      OBTAIN COS(L*PHI) AND SIN(L*PHI)
C
C      CL = STORE(JJJ,2*1+3)
C      SL = STORE(JJJ,2*1+4)
C      LL = L - 1
C      IF(L.EQ.0)LL = 1
C
C      RETRIEVE THE BESSEL FUNCTIONS OF ORDER L AND L-1
C
C      BK3 = CON1*BKST3(LL+1)
C      BK3M = CON1*BKST3(LL+1)
C      BJ = BJST(LL+1)
C      BJM = BJST(LL+1)
C      IF(L.EQ.0)BJM=-BJM
C
C      DO 100 J = 1,NOT
C      S = INDX(J)
C      XS = S
C
C      OBTAIN COS(S*PHI) AND SIN(S*PHI)
C
C      CS = STORE(JJJ,2*J+3)
C      SS = STORE(JJJ,2*J+4)
C
C      RETRIEVE THE BESSEL FUNCTIONS OF ORDER S AND S-1
C
C      LL = S - 1
C      IF(S.EQ.0)LL = 1
C      B12 = CON1*B1ST2(S+1)
C      B12M = CON1*B1ST2(LL+1)
C      B11 = B1ST1(S+1)
C      B11M = B1ST1(LL+1)
C      BK2 = CON1*BKST2(S+1)
C      BK2M = CON1*BKST2(LL+1)
C      BK1 = BKST1(S+1)
C      BK1M = BKST1(LL+1)
C
C      TN1 = AK2R2*B12M*BK3 + AK3R2*B12*BK3M
C      TN2 = B12*BK3
C      TN1 = -AK2R2*BK2M*BK3 + AK3R2*BK2*BK3M
C      TN2 = BK2*BK3

```

```

      101 = AK1R1*BJ*BJ*BI1 - AK2R1*BJ*BJ*BI1M
      102 = -BJ*BJ*BI1
      103 = AK1R1*BJ*BJ*BK1 + AK2R1*BJ*BJ*BK1M
      104 = -BJ*BJ*BK1

C
      IF(SC.EQ.2) GO TO 250
      ZETA1 = 0.0
      ZETA2 = 0.0
      IF (L.EQ.5) GO TO 80
      ZETA1 = (XL - XS)*CL*CS + (XS*CL*SS - XL*SL*CS)*DR1/RH1
      ZETA2 = (XL - XS)*CL*CS + (XS*CL*SS - XL*SL*CS)*DR2/RH2
      80 MLS = TM1*CL*CS + TM2*ZETA2
      NLS = TM1*CL*CS + TM2*ZETA2
      PLS = TP1*CL*CS + TP2*ZETA1
      QLS = TQ1*CL*CS + TQ2*ZETA1
      GO TO 91

C
      250 ZETA1 = 0.0
      ZETA2 = 0.0
      IF(L.EQ.5) GO TO 81
      ZETA1 = (XL - XS)*SL*SS + (XL*CL*SS - XS*SL*CS)*DR1/RH1
      ZETA2 = (XL - XS)*SL*SS + (XL*CL*SS - XS*SL*CS)*DR2/RH2
      81 MLS = TM1*SL*SS + TM2*ZETA2
      NLS = TM1*SL*SS + TM2*ZETA2
      PLS = TP1*SL*SS + TP2*ZETA1
      QLS = TQ1*SL*SS + TQ2*ZETA1

C
      91 GDM(I,J) = GDM(I,J) + MLS
      GDN(I,J) = GDN(I,J) + NLS
      GDP(I,J) = GDP(I,J) + PLS
      GDO(I,J) = GDO(I,J) + QLS
      100 CONTINUE

C
      C LOAD THE FOUR SUBMATRICES INTO GD
      DO 105 J = 1,NOT2
      DO 105 I = 1,NOT2
      IF(I.LE.NOT.AND.J.LE.NOT)GD(I,J) = GDM(I,J)
      IF(I.LE.NOT.AND.J.GT.NOT)GD(I,J) = GDN(I,J-NOT)
      IF(I.GT.NOT.AND.J.LE.NOT)GD(I,J) = GDP(I-NOT,J)
      IF(I.GT.NOT.AND.J.GT.NOT)GD(I,J) = GDO(I-NOT,J-NOT)
      GD(I,J) = GD(I,J)*W*CON2
      105 CONTINUE

C
      RETURN
      END
      SUBROUTINE PERIM(N,R,PHI,RHO,DRDP)
      C
      C EVALUATE RHO(PHI) AND D(RHO)/D(PHI) ON THE PERIMETER
      C
      C REAL N
      C
      C IF(R.NE.1.0.OR.N.NE.1.0)GO TO 100
      RHO = 1.0
      DRDP = 0.0
      RETURN

C
      100 CONTINUE
      CP = COS(PHI)
      SP = SIN(PHI)
      PN = 2.0*N
      DD = (CP/R)*PN + SP*PN
      PN = 1.0/PN
      RHO = 1.0/(DD*PN)
      C
      PN = 2.0*N - 2.0

```

```

      S = SP*OPN - ((CP/R)*OPN)/(R*OP)
      P = -RHO*SP*CP*AA/DO
      RETURN
END
SUBROUTINE MATIN(A,N,B,DETER,M)
  DIMENSION A(24,24),B(24,1),INDEX(24,3)
  DIMENSION DDD(24),PVT(24)
  COMMON/M/DDO,PVT
  DETER = 1.0
  DO 20 J = 1,N
    INDEX(J,3) = 0
  DO 550 I = 1,N
    SEARCH FOR PIVOT ELEMENT
    AMAX = 0.
    DO 105 J = 1,N
      IF (INDEX(J,3)-1) 60,105,60
    DO 100 K = 1,N
      IF (INDEX(K,3)-1) 80,100,740
    80 IF(AMAX-ABS(A(J,K))) 85,100,100
    85 IROW = J
    ICOL = K
    AMAX = ABS(A(J,K))
  100 CONTINUE
  105 INDEX(ICOL,3) = INDEX(ICOL,3) + 1
  INDEX(I,1) = IROW
  INDEX(I,2) = ICOL
  INTERCHANGE ROWS TO PUT PIVOT ELEM.IN DIAG.
  S = 1.0
  IF(IROW-ICOL) 140,310,140
  140 DETER = -DETER
  F = -1.0
  DO 200 L = 1,N
    SWAP = A(IROW,L)
    A(IROW,L) = A(ICOL,L)
    200 A(ICOL,L) = SWAP
    IF(M) 310,310,210
  210 DO 250 L = 1,M
    SWAP = B(IROW,L)
    B(IROW,L) = B(ICOL,L)
    250 B(ICOL,L) = SWAP
  C
  C
  DIVIDE PIVOT ROW BY PIVOT ELEMENT
  C
  310 PIVOT = A(ICOL,ICOL)
  DETER = DETER*PIVOT
  DDD(I) = DETER
  PVT(I) = PIVOT*S
  A(ICOL,ICOL) = 1.0
  DO 350 L = 1,N
    350 A(ICOL,L) = A(ICOL,L)/PIVOT
    IF(M) 380,380,360
  360 DO 370 L = 1,M
    370 B(ICOL,L) = B(ICOL,L)/PIVOT
  C
  C
  REDUCE NON PIVOTS ROWS
  C
  DO 550 I = 1,N
    IF(I1-ICOL) 400,550,40

```

```

400 I A(L1, ICOLUM)
      I, ICOLUM) = 0.
DU 450 L = 1, N
450 A(L1, L) = A(L1, L) - A(ICOLUM, L) * I
      IF(M) 550, 550, 460
460 DO 500 L = 1, M
500 B(L1, L) = B(L1, L) - B(ICOLUM, L) * I
550 CONTINUE
C
C INTERCHANGE COLUMNS
C
DD 710 I = 1, N
L = N + 1 - I
IF(INDEX(L, 1) - INDEX(L, 2)) 630, 710, 630
630 JROW = INDEX(L, 1)
      JCOLUM = INDEX(L, 2)
DO 705 K = 1, N
      SWAP = A(K, JROW)
      A(K, JROW) = A(K, JCOLUM)
      A(K, JCOLUM) = SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
C
C 750 DETER = 0.0
      RETURN
      END

```

Program Listing

- HICOAX -

(Reference: Vol. I, Section II.4)

```

PROGRAM HICQAK(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)

PROGRAM HICQAX FINDS THE MODES OF A DIELECTRIC WAVEGUIDE HAVING THE SHAPE
OF A RING OF ARBITRARY CROSS-SECTION. THE WEAKLY-GUIDING CONDITION IS
ASSUMED TO HOLD TRUE. THE INDEX OF REFRACTION OF THE RING IS GREATER
THAN THAT OF THE SURROUNDING REGIONS, INSIDE AND OUTSIDE. THE INDEX
IN EACH OF THESE IS ASSUMED TO BE THE SAME.

INTEGER S,SZ
REAL NL,NLL,N1,N2
DIMENSION PR(10),TABLE(51,2),INDEX(20)
DIMENSION STORE(200,28),INDX(1,2),GD(24,24),C(24)
DIMENSION ROOT(20,14),ROOM(20,3),IROOT(20,3)
COMMON/A/B,P2,P1,NOT,NOT2
COMMON/B/STORE,INDX,GD,C,SZ
DATA (PR(1),1,-.3)/10H CIRCLE /,10H ELLIPSE /,10H SQUARE /
DATA (PR(1),1,-4.7)/10H S' ELLIPSE /,10H RECTANGLE,10H COSINE,6H SINE /
DATA (PR(1),1,-8.9)/10H FLIPPED /,10H CUSPED /

*****

LIST OF VARIABLES WITH SINGLE ASSIGNED PURPOSES

ACC ACCURACY TO WITHIN WHICH THE ROOT-LOCATION PROCEDURE FINDS THE ROOTS
B NORMALIZED FREQUENCY (READ IN)
C VECTOR OF COEFFICIENTS IN THE FIELD EXPANSION (DESIRED RESULT)
DET VALUE OF THE DETERMINANT OF THE MATRIX GD
DROP D(RHO)/D(PHI) ON THE PERIMETER OF THE GUIDE
GD MATRIX OF COEFFICIENTS
ID # OF FUNCTION EVALUATIONS DURING ROOT-LOCATION PROCEDURE
IDIAG DIAGNOSIS CODE (SEE COMMENTS) (READ IN)
INDEX ARRAY FOR STORING TABLE INDICES NEAR WHICH ROOTS OCCUR
INDX ARRAY FOR STORING VALUES OF INDICES USED IN FIELD EXPANSION
IROOT ARRAY FOR STORING INFORMATION USED IN PRINTING RESULTS AT THE ROOT
JDEX COUNTER TO KEEP TRACK OF THE NUMBER OF SEPARATE CASES COMPLETED
L INDEX OF 1ST TERM IN FIELD EXPANSION (READ IN)
LLL VALUE OF L FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
LLS VALUE OF S FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
NL,N2 SUPERELLIPSE INDICES (REAL) (READ IN)
NL VALUE OF N1 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
NLL VALUE OF N2 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
NOP # OF INTEGRATION POINTS (READ IN)
NOT # OF TERMS IN EACH SUBMATRIX (READ IN)
NOT2 DIMENSION OF DETERMINANT = 2*NOT
NP2 # OF VALUES OF P2 TAKEN TO CONSTRUCT TABLE (READ IN)
NR # OF ROOTS FOUND
PHI ANGLE IN (RHO, PHI) COORDINATE SYSTEM
PI P1
P2 LARGEST VALUE OF P2 USED IN TABLE (READ IN)
PMAX SMALLEST VALUE OF P2 USED IN TABLE (READ IN)
PMIN ARRAY FOR STORING ALPHANUMERIC PROGRAM INFORMATION
PR PROPAGATION CONSTANT (DESIRED RESULT)
P2 R1,R2 ASPECT RATIOS (READ IN)
R1,R2
RHO RADIUS IN (RHO, PHI) COORDINATE SYSTEM
RH0 VALUE OF R1 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
RLL VALUE OF R2 FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
ROOM ARRAY FOR STORING OUTPUT INFORMATION
ROOT ARRAY FOR STORING OUTPUT INFORMATION, INCLUDING VALUES OF P2
S INDEX OF LAST TERM IN FIELD EXPANSION (INTEGER)
SC INDICATES SINE OR COSINE EXPANSION (INTEGER) (READ IN)
STORE ARRAY FOR STORING FUNCTIONS USED REPEATEDLY, WHICH DEPEND ONLY ON PHI
SZ RATIO OF INNER SEMIMAJOR AXIS TO OUTER (0.1E-52.LT.1)
SZL VALUE OF SZ FROM PREVIOUS CASE (HELPS AVOID REPEATED CALCULATIONS)
SZL

```



```

C      N= NUMBER OF INTERVALS IN THE INITIAL TABLE          DEF: 10
C      I= NUMBER OF INTERGRATION POINTS IN THE RANGE (0,PI/2) { DEF: 50
C      I= DIAGNOSIS CODE: 0 = NORMAL CONDITION (DEFAULT)
C      1 = INITIAL TABLE, FINAL MATRIX PRINTED
C      2 = SUMMARY PRINTED AT END (VALUES OF P2)
C      3 = DETAILS OF ROOT-LOCATION PRINTED
C
C      * IF THE CROSS-SECTION IS CIRCULAR, THE PHI-INTEGRALS CAN BE DONE IN
C      * CLOSED FORM. TO SAVE TIME, ONE CAN SET NOP = 1, LEAVING ONLY ONE "IN-
C      * TEGRATION POINT". IN THIS CASE ADJUSTMENTS IN THE VALUES OF THE TRIG
C      * FUNCTIONS ARE MADE AT STATEMENT 87 SO THAT THE INTEGRALS WILL COME OUT
C      * RIGHT. NOTE THAT IF NOP = 1, NOT MUST ALSO EQUAL 1 OR ELSE THE OFF-
C      * DIAGONAL MATRIX ELEMENTS MAY BE NONZERO AND THEREFORE INCORRECT.
C
C      READ IN INFORMATION WHICH REMAINS FIXED FOR MANY CASES.
C      RETURN TO STATEMENT 2 ONLY AFTER DUMPING RESULTS FOR THESE CASES.
C      A BLANK CARD HERE TERMINATES PROGRAM EXECUTION.
C
C      READ(5,100)NOT,$,NP2,NOP,IDIAG
C      IF(NOT.EQ.0)STOP
C      WRITE(6,130)
C      WRITE(6,123)
C      WRITE(6,101)PN,N
C      IF(NOT.GT.12)NOT = 12
C      IF($.LE.0)$ = 3
C      IF($.GT.12)$ = 12
C      ACC = 10.0*(1-$)
C      IF(NP2.LT.1)NP2 = 10
C      IF(NP2.GT.50)NP2 = 50
C      IF(NOP.GT.200)NOP = 200
C      IF(NOP.EQ.0)NOP = 50
C      IF(NOP.EQ.1)NOT = 1
C      NOT2 = 2*NOT
C
C      PRINT THE FIXED INFORMATION READ FROM THE FIRST DATA CARD
C
C      WRITE(6,102)NOT,NOT2
C      WRITE(6,104)$
C      WRITE(6,105)NP2
C      WRITE(6,108)NOP
C      WRITE(6,103)IDIAG
C
C      * * * * *
C      SC = 1 IF COSINE TERMS ARE TO BE USED; 2 IF SINE TERMS ARE DESIRED
C      L = INDEX OF FIRST TERM IN FIELD EXPANSION
C      N1,N2 = INDICES OF OVOIDS (1 = ELLIPSE, >30 = SQUARE)
C      N1,N2 = INDICES OF OVOIDS (1 = ELLIPSE, >30 = SQUARE)
C      R1,R2 = ASPECT RATIOS
C      SZ = RATIO OF SEMINOR AXES, R1/R2 (0 < SZ < 1)
C      B = ASSUMED VALUE OF THE BINDING CONSTANT
C      PMAX, PMIN = RANGE OF VALUES OF P2 TO BE SEARCHED
C      DEF: 1
C      DEF: 1
C      DEF: 1
C      DEF: 1, 0
C
C      ONE DATA CARD CORRESPONDS TO ONE INDIVIDUAL CASE.
C
C      RETURN TO STATEMENT 1 IMMEDIATELY AFTER EACH CASE IS COMPLETE. A BLANK
C      CARD READ AT STATEMENT 1 CAUSES RESULTS FOR ALL CASES TO BE DUMPED
C      (PRINTED) AND CONTROL WILL THEN GO TO STATEMENT 2. A BLANK CARD THERE
C      TERMINATES THE JOB EXECUTION, SO TWO BLANK CARDS SHOULD FOLLOW THE LAST
C      DATA CARD.
C
C      JDEX = 0
C      1 READ(5,121)$C,L,N1,R1,N2,R2,SZ,B,PMAX,PMIN
C      IF(R1.LE.0.0)GO TO 999
C      IF(JDEX.GE.20)GO TO 999
C      JDEX = JDEX + 1

```



```

96 ( TO 92
    I=2 (N1.NE.1.0) GO TO 92
    IF (R1.EQ.1.0) I=1
90 IF (N1.LT.30.) GO TO 92
    I=5
    IF (R1.EQ.1.0) I=3
92 CONTINUE
    W = PR(I)
    IF(NOP.EQ.1.AND.I.NE.1)WRITE(6,131)
    I=4
    IF (N2.GT.1.0) GO TO 50
    IF (N2-.5) 54,55,56
54 I=9
    GO TO 52
55 I=8
    GO TO 52
56 IF (N2.NE.1.0) GO TO 52
    I=2
    IF (R2.EQ.1.0) I=1
50 IF (N2.LT.30.) GO TO 52
    I=5
    IF (R2.EQ.1.0) I=3
52 CONTINUE
    WRITE(6,112)PR(J),W,PR(I)
    IF(NOP.EQ.1.AND.I.NE.1)WRITE(6,131)
    ROOT(JDEX,2) = PR(J)
    ROOT(JDEX,1) = W
    ROOM(JDEX,1) = PR(I)

```

C
C
C

```

PN = NOP
W = PI/(PN+2.0)
IF(RL.NE.R1.OR.RLL.NE.R2)GO TO 71
IF(NL.NE.N1.OR.NLL.NE.N2)GO TO 71
IF(SZ.NE.SZL)GO TO 71
GO TO 79
71 CONTINUE
RLL = R2
NLL = N2
RL = R1
NL = N1
SZL = SZ

```

C
C
C

CALCULATE AND STORE RHO(PHI) AND D(RHO)/D(PHI) FOR EACH BOUNDARY

```

DO 75 I = 1,NOP
PN = 2*(NOP - I) + 1
PHI = W*PN/2.0
CALL PERIM(N1,R1,PHI,RHO,DRDP)
STORE(I,1) = RHO*SZ
STORE(I,2) = DRDP*SZ
CALL PERIM(N2,R2,PHI,RHO,DRDP)
STORE(I,3) = RHO
STORE(I,4) = DRDP
75 STORE(I,4) = DRDP

```

C
C
C

CALCULATE AND STORE COS(M*PHI) AND SIN(M*PHI) WHERE M CAN BE L,.....S.
VALUES OF M ARE STORED AS INDX.

```

79 CONTINUE
IF(L.EQ.LLL.AND.S.EQ.LLS)GO TO 89
LLL = L
LLS = S
INDX(1) = L
IF(NOT.EQ.1)GO TO 86

```

C

```

      T = (S - L)/(NOT - 1)
      95 K = 2, NOT
      96 INDEX(K) = L + (K - 1)*LST
      IF(NOP.EQ.1)GO TO 87
      DO 80 I = 1, NOP
      PN = 2*(NOP - 1) + 1
      PHI = M*PN/2.0
      DO 80 M = 5, LST, 2
      K = M/2 - 1
      PN = INDEX(K)
      PN = PN*PHI
      STORE(I, M) = COS(PN)
      80 STORE(I, M+1) = SIN(PN)
      GO TO 89
C
C IF NOP = 1, ASSUME CIRCULAR CROSS-SECTIONS. SET TRIG FUNCTIONS EQUAL TO
C CONSTANTS CHOSEN SO THAT THE ANALYTICAL RESULTS FOR THE INTEGRALS ARE
C OBTAINED.
C
      87 STORE(1, 5) = SORT(.5)
      STORE(1, 6) = SORT(.5)
      IF(INDEX(1).NE.0)GO TO 89
      STORE(1, 5) = 1.0
C
C .....
C
C CONSTRUCT THE TABLE OF DETERMINANT VALUES FOR PMIN < P2 < PMAX.
C FOR EACH VALUE OF P2. SUBROUTINE CHEBY CALCULATES THE MATRIX
C GD, WHOSE DETERMINANT IS SOUGHT. SUBROUTINE MATINV CALCULATES
C THE DETERMINANT.
C
      89 LST = NP2 + 1
      IF(IDIAG.GT.0)WRITE(6,106)
      PN = NP2
      PN = (PMAX - PMIN)/PN
      M = PN/100.
      P2 = PMAX
      IF(PMAX.EQ.1.0)P2 = 1.0 - M
      DO 200 M = 1, LST
      CALL CHEBY(NOP, SC)
      CALL MATINV(GD, NOT2, C, DET, 0)
      TABLE(M, 1) = P2
      TABLE(M, 2) = DET
      IF(IDIAG.GT.0)WRITE(6,107)M, P2, DET
      P2 = PMAX - M*PN
      200 IF(P2.LT.M)P2 = M
      IF(IDIAG.GT.0)WRITE(6,114)
C
      CALL SECOND(T1)
      T1 = T1 - TO
      WRITE(6,115)T1
C
C .....
C
C MAKE TENTATIVE IDENTIFICATION OF ROOTS OF EQUATION DET(RLS) = 0
C
      NR = 0
      IF(TABLE(1, 2).NE.0.0)GO TO 210
      NR = 1
      INDEX(NR) = 1001
      210 DO 220 M = 2, LST
      PN = ABS(TABLE(M, 2))
      IF(PN.GT.0.0)GO TO 215
      NR = NR + 1
      INDEX(NR) = M + 1000

```

```

GO TO 220
215 ( = (TABLE(M,2)/PN)*TABLE(M-1,2)
      PN,GE,0,0)GO TO 220
      NR = NR + 1
      INDEX(NR) = M - 1
220 IF(NR.GE.20)GO TO 221
221 WRITE(6,124)NR,JDEX
      IF(NR.EQ.0)GO TO 1
      .....
      LOCATE THE ROOTS OF DET(RLS) PRECISELY.
      LOOP DO 250 IS COMPLETED ONCE FOR EACH ROOT (I) IDENTIFIED ABOVE
      IF THERE ARE MORE THAN 8 ROOTS, MAKE NOTE OF IT. ONLY THE FIRST 8
      ROOTS WILL BE LISTED IN THE SUMMARY AT THE END. (IF THERE ARE MORE
      THAN 20 ROOTS, THE LAST ONES WILL HAVE BEEN MISSED COMPLETELY.)
      K = NR
      IF(NR.LE.8)GO TO 251
      K = K - 8
      WRITE(6,128)K
      K = 8
251 IROOT(JDEX,1) = K
      DO 250 I = 1,NR
      M = INDEX(I)
      CALL RTT(TABLE,ACC,PN,NOP,SC,M,IO,IDIAG)
      RTT RETURNS A FINAL MATRIX, GD, A SET OF COEFFICIENTS, C, AND THE
      ROOT, P2. SEE THE SUBROUTINE FOR A DESCRIPTION OF PARAMETERS.
      PRINT THE MATRIX, IF DESIRED; STORE ROOT; CALCULATE A FINAL VALUE
      FOR THE DETERMINANT. ALSO PRINT THE COEFFICIENTS.
      IF(I.LE.8)ROOT(JDEX,I+6) = P2
      IF(IDIAG.LT.1)GO TO 253
      W = 3HGD
      WRITE(6,114)
      DO 252 J = 1,NOT2
252 WRITE(6,118)W,(GD(J,K),K=1,NOT2)
253 WRITE(6,114)
      CALL MATINV(GD,NOT2,C,DET,0)
      WRITE(6,125)JDEX,I,P2,PN,IO,DET
      PN = 1H
      WRITE(6,122)((PN,INDEX(K),C(K)),K=1,NOT)
      J = NOT + 1
      WRITE(6,132)((PN,INDEX(K-NOT),C(K)),K=J,NOT2)
      WRITE(6,114)
      IF(DET.EQ.0.0)WRITE(6,113)
250 CONTINUE
      CALL SECOND(T1)
      T1 = T1 - TO
      WRITE(6,126)T1
      GO TO 1
      .....
      AFTER ALL CASES (20 OR LESS) HAVE BEEN COMPLETED, PRINT A TABLE
      OF RESULTS GIVING THE ROOTS FOR EACH CASE.
      999 WRITE(6,123)
          WRITE(6,123)
          WRITE(6,123)

```

```

WRITE(6,116)
C 900 I = 1,JDEN
C IROOT(I,1) = 6
C IF(IDIAG.LE.1)WRITE(6,117)1,ROOT(1,1),1,ROOT(1,M),M=2,3)
C 1,ROOT(1,J),J=2,K)
C IF(IDIAG.GE.2)WRITE(6,127)1,ROOT(1,1),1,ROOT(1,M),M=2,3)
C 1,ROOT(1,J),J=2,K)
C 900 WRITE(6,129)(ROOM(1,M),M=1,3)
C GO TO 2
C END
C SUBROUTINE RTT(TABLE,ACC,PN,NOP,SC,M,IO,IDIAG)
C
C SUBROUTINE RTT PRECISELY LOCATES THE ROOT SPECIFIED BY M (USING
C ELEMENTS M AND M+1 IN THE ARRAYS TABLE(M,1) AND TABLE(M,2)), AND THEN
C DETERMINES THE FIELD EXPANSION COEFFICIENTS C(J) CORRESPONDING TO
C THAT ROOT. ACC IS THE DESIRED ACCURACY OF THE ROOT, AND PN IS THE
C DIFFERENCE IN THE ACTUAL SOUNDS OBTAINED. IO IS THE NUMBER OF FUN-
C CTION EVALUATIONS NECESSARY. SC, IDIAG, AND NOP ARE AS IN MAIN. GO, THE
C FINAL MATRIX AT THE ROOT P2, IS ALSO RETURNED. (C, GO, AND P2 ARE ALL
C IN COMMON BLOCKS.)
C
C INTEGER SC
C DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24),SINV(24,24)
C DIMENSION TABLE(51,2)
C COMMON/A/B,P2,P1,NOT,NOT2
C COMMON/B/STORE,INDX,GD,C,SZ
C
C 700 FORMAT(/,*, SEARCH IO=7K,*,11X,*,11X,*,10X,*,F10.9X,*,F20.11X,*,DX*,
C 19X,*,P2=,9X,*,F*)
C 701 FORMAT(/,*, $$$$ ERROR IN HINTS OR REQ, IER =,13,*, $$$$$,/)
C
C IF(IDIAG.GE.3)WRITE(6,700)
C K = 0
C ID = 0
C IF(M.LT.1000)GO TO 230
C P2 = TABLE(M-1000,1)
C CALL CHEBY(NOP,SC)
C ID = 1
C GO TO 248
C
C 230 G1 = TABLE(M+1,2)
C G2 = TABLE(M,2)
C X1 = TABLE(M+1,1)
C X2 = TABLE(M,1)
C DN = X2 - X1
C
C IF G1 AND G2 ARE TWO OR MORE ORDERS OF MAGNITUDE DIFFERENT, USE
C A HALF-INTERVAL SEARCH TO NARROW THE RANGE X1 TO X2
C
C 235 W = -G1/G2
C IF(W.LT.100..AND.W.GT..01)GO TO 240
C IF(DN.LT.ACC)GO TO 240
C DN = ON/3.0
C
C CALL HINTS(X1,X2,G1,G2,DN,PN,NOP,SC,IO,IDIAG,IER)
C IF(1ER.GT.0)WRITE(6,701)IER
C
C K = K + IO
C IF(PN.EQ.0.0)GO TO 248
C GO TO 235
C
C MODIFIED REGULA-FALSI METHOD TO GET THE ROOT PRECISELY
C
C 240 CALL REG(X1,X2,G1,G2,ACC,PN,NOP,SC,IO,IDIAG,IER)
C IF(1ER.GT.0)WRITE(6,701)IER

```

```

C 248 ( = ID + K
C
C CALCULATE THE COEFFICIENTS. SUBROUTINE LOAD COPIES (NOT2-1) OF THE
C NOT2 ROWS OF GO INTO ARRAY SINV AND VECTOR C. SUBROUTINE MATINV SOLVES
C THE (NOT2-1)X(NOT2-1) SYSTEM FOR THE COEFFICIENTS AND RETURNS THEM IN C.
C
C CALL LOAD(GO,SINV,C,NOT2,M)
C LST = NOT2 - 1
C CALL MATINV(SINV,LST,C,DN,1)
C
C STACK THE COEFFICIENTS IN THE PROPER ORDER IN VECTOR C, INCLUDING 1.000
C
C IF(M.EQ.NOT2)GO TO 280
C DO 270 J = M,LST
C K = NOT2 + M - J
C 270 C(K) = C(K-1)
C 280 C(M) = 1.0
C RETURN
C END
C SUBROUTINE LOAD(SS,SINV,C,NOT2,ID)
C
C SUBROUTINE LOAD LOADS SS (NEE GO) INTO ARRAY SINV, ELIMINATING
C THE ROW AND COLUMN CONTAINING THE DIAGONAL ELEMENT WHICH IS SMALLEST
C IN MAGNITUDE. THE NEGATIVE OF THE COLUMN OF SS WHICH IS LEFT OUT IS
C LOADED INTO ARRAY C, EXCEPT FOR THE ELEMENT IN THE ROW WHICH IS ELIM-
C INATED. THUS, SINV IS (NOT2-1) BY (NOT2-1) IN DIMENSION, AND C IS A VEC-
C TOR (NOT2-1) IN LENGTH.
C
C NOTE THAT ONLY THE FIRST THREE DIAGONAL ELEMENTS ARE SEARCHED
C FOR THE SMALLEST ONE.
C
C DIMENSION SS(24,24),SINV(24,24),C(24)
C G = 1.0E+321
C ID = 1
C K = NOT2
C IF(K.GT.3)K = 3
C DO 200 J = 1,K
C H = ABS(SS(J,J))
C IF(H.GT.G)GO TO 200
C G = H
C ID = J
C 200 CONTINUE
C
C JJ = 0
C DO 300 J = 1,NOT2
C IF(J.EQ.ID)GO TO 300
C JJ = JJ + 1
C C(JJ) = -SS(J,ID)
C KK = 0
C DO 299 K = 1,NOT2
C IF(K.EQ.ID)GO TO 299
C KK = KK + 1
C SINV(JJ,KK) = SS(J,K)
C 299 CONTINUE
C 300 CONTINUE
C RETURN
C END
C SUBROUTINE HINTS(X1,X2,G1,G2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C
C SUBROUTINE HINTS PERFORMS A HALF-INTERVAL SEARCH TO LOCATE A ROOT
C OF THE EQUATION DET(RLS) = 0 TO WITHIN A DESIRED ACCURACY ACC.
C ID IS THE NUMBER OF BISECTIONS, AND PN IS THE ACTUAL MAXIMUM ERROR.
C
C INTEGER SC
C DIMENSION STORE(200,28),INDX(12),GO(24,24),C(24)

```

```

C      (  WCON/A/B,P2,P1,NOT,NOT2
          COMMON/B/STORE,INDX,GD,C,SZ
          ID = 0
          IER = 0
          100 PN = ABS(X2 - X1)
C      IF(PN.LT.ACC)RETURN
          ID = ID + 1
          IF(ID.GT.25)GO TO 230
          P2 = (X1 + X2)/2.0
          CALL CHEBY(NOP,SC)
          CALL MATINV(GD,NOT2,C,G,0)
C      IF(IDIAG.GE.3)WRITE(6,700)ID,X1,X2,G1,G2,PN,P2,G
          700 FORMAT(' HINTS:',I3.2F13.10,3E11.4,F13.10,E11.4)
C      IF(G*(G1/ABS(G1)))120,210,130
          120 X2 = P2
              G2 = G
              GO TO 100
          130 X1 = P2
              G1 = G
              GO TO 100
C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE BEEN
C      ALTERED IN MATINV) IS RECOMPUTED.
C      210 PN = 0.0
          CALL CHEBY(NOP,SC)
          ID = ID + 1
          RETURN
C      ERROR RETURN: TOO MANY ITERATIONS
C      230 IER = 6
          RETURN
          END
C      SUBROUTINE REG(X1,X2,F1,F2,ACC,PN,NOP,SC,ID,IDIAG,IER)
C      SUBROUTINE REG SOLVES THE EQUATION DET(RLS) = 0 FOR P2 TO WITHIN
C      A DESIRED ACCURACY ACC BY THE MODIFIED REGULA-FALSI METHOD. ID IS
C      THE NUMBER OF FUNCTION EVALUATIONS AND PN GIVES THE ACTUAL MAXIMUM ERROR.
C      INTEGER SC
          DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24)
          COMMON/A/B,P2,P1,NOT,NOT2
          COMMON/B/STORE,INDX,GD,C,SZ
C      ID = 0
          IER = 0
          G = F1
          100 PN = ABS(X1 - X2)
C      F0 = G/ABS(G)
          P2 = (F2*X1 - F1*X2)/(F2 - F1)
          CALL CHEBY(NOP,SC)
          ID = ID + 1
          IF(PN.LT.ACC)RETURN
          IF(ID.GT.25)GO TO 230
          CALL MATINV(GD,NOT2,C,G,0)
C      IF(IDIAG.GT.2)WRITE(6,700)ID,X1,X2,F1,F2,PN,P2,G
          700 FORMAT(' REG: ',I3.2F13.10,3E11.4,F13.10,E11.4)
C      IF(G*(F1/ABS(F1)))150,190,160

```

```

150      F = P2
      G = G
      A = ((F0*G).GT.0.0)F1 = F1/2.0
      GO TO 100
160      X1 = P2
      F1 = G
      IF((F0*G).GT.0.0)F2 = F2/2.0
      GO TO 100
C
C      IF THE DETERMINANT IS EXACTLY ZERO, THE MATRIX (WHICH WILL HAVE
C      BEEN ALTERED IN MATINV) IS RECOMPUTED.
C
190      PN = 0.0
      CALL CHEBY(NOP,SC)
      ID = ID + 1
      RETURN
C
C      ERROR RETURN: TOO MANY ITERATIONS
C
230      IER = 5
      RETURN
      END
C
C      SUBROUTINE CHEBY(NOP,SC)
C
      INTEGER S,SC
      REAL NLS,NLS
      DIMENSION STORE(200,28),INDX(12),GD(24,24),C(24)
      DIMENSION GDM(12,12),GDM(12,12),GDP(12,12),GDD(12,12)
      DIMENSION BIST1(49)
      DIMENSION BKST3(49),BYST1(49),BYST2(49),BYST2(49)
      COMMON/A/B,P2,P1,NOT,NOT2
      COMMON/B/STORE,INDX,GD,C,SZ
C
C      CHEBYSHEV INTEGRATION OF THE MATRIX ELEMENTS OF RLS
C
      NMAX = INDX(NOT) + 1
      IF(NMAX.EQ.1)NMAX = 2
      OR = 0.0
      NN = 0
      PTS = NOP
      W = PI/(PTS*2.0)
      P = SORT(P2)
      OMP = SORT(1.0 - P2)
      DO 50 I = 1,NOT
      DO 50 J = 1,NOT
      GDM(I,J) = 0.0
      GDN(I,J) = 0.0
      GDP(I,J) = 0.0
      GDD(I,J) = 0.0
50      GDD(I,J) = 0.0
C
C      LOOP DO 100 DOES INTEGRATION USING NOP POINTS IN THE RANGE (0,PI/2)
C
      DO 100 JJJ = 1,NOP
C
      OBTAIN RHO(PHI), COS(PHI), AND SIN(PHI)
C
      RH1 = STORE(JJJ,1)
      RH2 = STORE(JJJ,3)
      DR1 = STORE(JJJ,2)
      DR2 = STORE(JJJ,4)
C
      AK1R1 = K1*RHO1 FOR SURFACE 1-2
      AK2R1 = K2*RHO1 FOR SURFACE 1-2
      AK2R2 = K2*RHO2 FOR SURFACE 2-3
      AK3R2 = K3*RHO2 FOR SURFACE 2-3
C

```



```

C      R1 = PI*B*RH1*SQRT(P2)
C      R2 = PI*B*RH1*SQRT(1.0 - P2)
C      AK3R2 = PI*B*RH2*SQRT(P2)
C      AK2R2 = PI*B*RH2*SQRT(1.0 - P2)
C      OBTAIN THE BESSEL FUNCTIONS WHICH WILL BE REQUIRED
C
C      IL(AK1R1) IS STORED IN B1ST1
C      JS(AK2R1) IS STORED IN B1ST1
C      JS(AK2R2) IS STORED IN B1ST1
C      KL(AK3R2) IS STORED IN B1ST1
C      KL(AK3R2) IS STORED IN B1ST1
C      YS(AK2R1) IS STORED IN B1ST1
C      YS(AK2R2) IS STORED IN B1ST1
C
C      CALL BESJ(AK1R1,OR,1,NMAX,B1ST1,NZ)
C      CALL BESJ(AK2R1,OR,NMAX,B1ST1,NZ)
C      CALL BESJ(AK2R2,OR,NMAX,B1ST1,NZ)
C      CALL BESN(AK3R2,NN,1,NMAX,BKST3,NZ)
C      IF(SZ.GT.0.0)CALL BESYN(AK2R1,NN,NMAX,B1ST1)
C      CALL BESYN(AK2R2,NN,NMAX,B1ST2)
C
C      IF SZ = 0, THE Y BESSEL FUNCTIONS MUST BE DEFINED. ALTHOUGH USED IN
C      HUSB AND GLSB, THE LATTER ARE REDEFINED LATER.
C
C      IF(SZ.GT.0.0)GO TO 70
C      DO 60 I = 1,NMAX
C      60 B1ST1(I) = 0.0
C
C      LOOP DO 100 I... CORRESPONDS TO L
C      LOOP DO 100 J... CORRESPONDS TO S
C
C      70 DO 100 I = 1,NOT
C      L = INDX(I)
C      XL = L
C
C      OBTAIN COS(L*PHI) AND SIN(L*PHI)
C
C      CL = STORE(JJJ,2*I+3)
C      SL = STORE(JJJ,2*I+4)
C      LL = L - 1
C      IF(L.EQ.0)LL = 1
C
C      RETRIEVE THE BESSEL FUNCTIONS OF ORDER L AND L-1
C
C      BK3 = BKST3(LL+1)
C      BK3M = BKST3(LL+1)
C      B1 = B1ST1(LL+1)
C      B1M = B1ST1(LL+1)
C
C      DO 100 J = 1,NOT
C      S = INDX(J)
C      XS = S
C
C      OBTAIN COS(S*PHI) AND SIN(S*PHI)
C
C      CS = STORE(JJJ,2*J+3)
C      SS = STORE(JJJ,2*J+4)
C
C      RETRIEVE THE BESSEL FUNCTIONS OF ORDER S AND S-1
C
C      LL = S - 1
C      IF(S.EQ.0)LL = 1
C      B1 = B1ST1(S+1)
C      B1M = B1ST1(LL+1)
C      B1J2 = B1ST2(S+1)

```

```

      BU2M= BUST2(LL+1)
      (  = BUST1(S+1)
      )  = BUST1(LL+1)
      BY2 = BUST2(S+1)
      BY2M= BUST2(LL+1)
      IF(S.GT.0)GO TO 80
      BU1M = -BU1M
      BU2M = -BU2M
      BY1M = -BY1M
      BY2M = -BY2M
C
C 80 CONTINUE
      FLS = 2.0*B-RH2*(BK3*BU2M*OMP + P*BU2*BK3M)
      FLSB = 2.0*B-RH2*(BK3*BY2M*OMP + P*BY2*BK3M)
      MLS = PI*B-RH1*(BI*BU1M*OMP - P*BI*BUI)
      MLSB = PI*B-RH1*(BI*BY1M*OMP - P*BI*BY1)
      ELS = 0.0
      ELSB = 0.0
      GLS = 0.0
      GLSB = 0.0
C
C ELS, ELSB, GLS, AND GLSB ARE NOT ACTUALLY ZERO IF L = S, BUT THEY
C ARE MULTIPLIED BY ETA, WHICH IS ZERO IN THAT CASE
C
      IF(L.EQ.S)GO TO 85
      FLS = FLS + (XL - XS)*2.0*BK3*BU2/PI
      FLSB = FLSB + (XL - XS)*2.0*BK3*BY2/PI
      ELS = 2.0*BK3*BU2*DR2/(PI*RM2)
      ELSB = 2.0*BK3*BY2*DR2/(PI*RM2)
      IF(SZ.EQ.0)GO TO 85
      MLS = MLS + (XL - XS)*BI*BU1
      MLSB = MLSB + (XL - XS)*BI*BY1
      GLS = BI*BU1*DR1/RH1
      GLSB = BI*BY1*DR1/RH1
C
C 85 CONTINUE
      IF(SZ.GT.0)GO TO 87
      GLSB = 0.0
      MLSB = 0.0
      IF(L.NE.S)GO TO 87
      MLSB = 2.0*PI
      IF(L.EQ.0)GO TO 87
      MLSB = 2.0*((P/OMP)*L)/PI
C
C 87 CONTINUE
      IF(SC.EQ.2)GO TO 90
      ETA = XS*CL*SS - XL*SL*CS
      MLS = FLS*CL*CS + ELS*ETA
      MLS = FLSB*CL*CS + ELSB*ETA
      PLS = MLS*CL*CS + GLS*ETA
      QLS = MLSB*CL*CS + GLSB*ETA
      GO TO 91
C
C 90 CONTINUE
      ETA = XS*SL*CS - XL*CL*SS
      MLS = FLS*SL*SS - ELS*ETA
      MLS = FLSB*SL*SS - ELSB*ETA
      PLS = MLS*SL*SS - GLS*ETA
      QLS = MLSB*SL*SS - GLSB*ETA
C
C 91 GDM(I,J) = GDM(I,J) + MLS
      GDM(I,J) = GDM(I,J) + MLS
      GDP(I,J) = GDP(I,J) + PLS
      GDQ(I,J) = GDQ(I,J) + QLS
C
C 100 CONTINUE

```

```

C      LOAD THE FOUR SUBMATRICES INTO GO
C      (
      DO 105 J = 1,NOT2
      DO 105 I = 1,NOT2
      IF(I.LE.NOT.AND.J.LE.NOT)GD(I,J) = GOM(I,J)
      IF(I.LE.NOT.AND.J.GT.NOT)GD(I,J) = GOM(I,J-NOT)
      IF(I.GT.NOT.AND.J.LE.NOT)GD(I,J) = GOM(I-NOT,J)
      IF(I.GT.NOT.AND.J.GT.NOT)GD(I,J) = GOM(I-NOT,J-NOT)
      GD(I,J) = GD(I,J)*W
105 CONTINUE
C      RETURN
C      END
C      SUBROUTINE PERIM(N,R,PHI,RHO,DRDP)
C      EVALUATE RHO(PHI) AND D(RHO)/D(PHI) ON THE PERIMETER
C      REAL N
C      IF(R.NE.1.0.OR.N.NE.1.0)GO TO 100
C      RHO = 1.0
C      DRDP = 0.0
C      RETURN
C      100 CONTINUE
C      CP = COS(PHI)
C      SP = SIN(PHI)
C      PN = 2.0*N
C      DD = (CP/R)*PN + SP*PN
C      PN = 1.0/PN
C      RHO = 1.0/(DD*PN)
C      PN = 2.0*N - 2.0
C      AA = SP*PN - ((CP/R)*PN)/(R*R)
C      DRDP = -RHO*SP*CP*AA/DD
C      RETURN
C      END
C      SUBROUTINE MATINV(A,N,B,DETER,M)
C      DIMENSION A(24,24),B(24,1),INDEX(24,3)
C      10 DETER = 1.0
C      15 DO 20 J = 1,N
C      20 INDEX(J,3) = 0
C      30 DO 550 I = 1,N
C      SEARCH FOR PIVOT ELEMENT
C      40 AMAX = 0.
C      45 DO 105 J = 1,N
C      IF (INDEX(J,3)-1) 60,105,60
C      60 DO 100 K = 1,N
C      IF (INDEX(K,3)-1) 80,100,740
C      80 IF (AMAX-ABS(A(J,K))) 85,100,100
C      85 IROW = J
C      90 ICOLUM = K
C      AMAX = ABS(A(J,K))
C      100 CONTINUE
C      105 CONTINUE
C      IF (AMAX.EQ.0.0)GO TO 750
C      INDEX(ICOLUM,3) = INDEX(ICOLUM,3) + 1
C      260 INDEX(I,1) = IROW
C      270 INDEX(I,2) = ICOLUM
C      INTERCHANGE ROWS TO PUT PIVOT ELEM.IN DIAG.
C      130 IF (IROW-ICOLUM) 140,310,140

```

```

140 DETER = -DETER
150 DO 200 L = 1,N
160   P = A(IROW,L)
170   A(IROW,L) = A(ICOLUMN,L)
200   A(ICOLUMN,L) = SWAP
      IF(M) 310,310,210
210 DO 250 L = 1,M
220   SWAP = B(IROW,L)
230   B(IROW,L) = B(ICOLUMN,L)
250   B(ICOLUMN,L) = SWAP
C
C   DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
310 PIVOT = A(ICOLUMN,ICOLUMN)
   DETER = DETER*PIVOT
330 A(ICOLUMN,ICOLUMN) = 1.0
340 DO 350 L = 1,N
350   A(ICOLUMN,L) = A(ICOLUMN,L)/PIVOT
355 IF (M) 380,380,360
360 DO 370 L = 1,M
370   B(ICOLUMN,L) = B(ICOLUMN,L)/PIVOT
C
C   REDUCE NON PIVOTS ROWS
C
380 DO 550 L1 = 1,N
390 IF (L1-ICOLUMN) 400,550,400
400   T = A(L1,ICOLUMN)
420   A(L1,ICOLUMN) = 0.
430 DO 450 L = 1,N
450   A(L1,L) = A(L1,L)-A(ICOLUMN,L)*T
455 IF (M) 550,550,460
460 DO 500 L = 1,M
500   B(L1,L) = B(L1,L)-B(ICOLUMN,L)*T
550 CONTINUE
C
C   INTERCHANGE COLUMNS
C
600 DO 710 I = 1,N
610   L = N + 1 - I
620 IF (INDEX(L,1)-INDEX(L,2)) 630,710,630
630   JROW = INDEX(L,1)
640   JCOLUMN = INDEX(L,2)
650 DO 705 K = 1,N
660   SWAP = A(K,JROW)
670   A(K,JROW) = A(K,JCOLUMN)
700   A(K,JCOLUMN) = SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
C
750 DETER = 0.0
   RETURN
   END

```

Program Listing

- VAR -

(Reference: Vol. I, Section II.5)

138

```

C      A = INNER-CORE RADIUS
C      (
C      = NORMALIZED FREQUENCY
C      PS1 = P2 OF UNPERTURBED INNER-CORE MODE
C      PS2 = P2 OF UNPERTURBED OUTER-CORE MODE
C      IDIAG = DIAGNOSTIC CODE
C
1 READ(5,101)L,A,C,B,PS1,PS2,IDIAG
  WRITE(6,120)
  IF(A.EQ.0.0)GO TO 900
  IF(JDEX.GE.40)GO TO 900
  JDEX = JDEX + 1
C
C      STORE INPUT VALUES FOR USE IN TABLE AT THE END
C
C      STORE(JDEX,1) = A
C      STORE(JDEX,2) = C
C      STORE(JDEX,3) = B
C      STORE(JDEX,4) = PS1
C      SSTORE(JDEX,1) = PS2
C      ISS(JDEX) = L
C
C      AL1 = PI*B*SQRT(1.0 - PS1)
C      AL2 = PI*B*SQRT(1.0 - PS2)
C      BT1 = PI*B*SQRT(PS1)
C      BT2 = PI*B*SQRT(PS2)
C      EPS = 1.0
C      IF(L.EQ.0)EPS = 2.0
C      WRITE(6,102)L,A,C,B
C      WRITE(,103)PS1,PS2,AL1,AL2,BT1,BT2
C
C      CALCULATE AND STORE THE BESSEL FUNCTIONS WHICH WILL BE NEEDED DURING
C      THE REMAINING PARTS OF THE PROGRAM
C
ORD = 0.0
NOR = 0
ARG = AL1*A
CALL BESJ(ARG,ORD,NUM,BJ1A,NZ)
ARG = BT1*A
CALL BESKN(ARG,NOR,1,NUM,BK1A,NZ)
ARG = BT1*C
CALL BESKN(ARG,NOR,1,NUM,BK1C,NZ)
ARG = BT1*D
CALL BESKN(ARG,NOR,1,NUM,BK1D,NZ)
C
ARG = AL2*C
CALL BESJ(ARG,ORD,NUM,BJ2C,NZ)
CALL BESYN(ARG,NOR,NUM,BY2C)
ARG = AL2*D
CALL BESJ(ARG,ORD,NUM,BJ2D,NZ)
CALL BESYN(ARG,NOR,NUM,BY2D)
ARG = BT2*A
CALL BESJ(ARG,ORD,1,NUM,BI2A,NZ)
ARG = BT2*C
CALL BESJ(ARG,ORD,1,NUM,BI2C,NZ)
CALL BESKN(ARG,NOR,1,NUM,BK2C,NZ)
ARG = BT2*D
CALL BESKN(ARG,NOR,1,NUM,BK2D,NZ)
C
C      STACK THE BESSEL FUNCTIONS (ORDERS -1, 0, 1, 2,...) IN POSITIONS 1, 2, 3,...
C      OF THE STORAGE VECTORS SO THAT, FOR EXAMPLE, JL(K1A) = BJ1A(L+2) FOR ALL L
C
DO 180 I = 1,NUM
  J = NUM - I + 1
  BJ1A(J+1) = BJ1A(J)

```

```

      R=1A(J+1) = BK1A(J)
      C(J+1) = BK1C(J)
      L=1D(J+1) = BK1D(J)
C
      BJ2C(J+1) = BJ2C(J)
      BJ2D(J+1) = BJ2D(J)
      BY2C(J+1) = BY2C(J)
      BY2D(J+1) = BY2D(J)
      B12A(J+1) = B12A(J)
      B12C(J+1) = B12C(J)
      BK2C(J+1) = BK2C(J)
      BK2D(J+1) = BK2D(J)
180 C
      BJ1A(1) = -BJ1A(3)
      BK1A(1) = BK1A(3)
      BK1C(1) = BK1C(3)
      BK1D(1) = BK1D(3)
C
      BJ2C(1) = -BJ2C(3)
      BJ2D(1) = -BJ2D(3)
      BY2C(1) = -BY2C(3)
      BY2D(1) = -BY2D(3)
      B12A(1) = B12A(3)
      B12C(1) = B12C(3)
      BK2C(1) = BK2C(3)
      BK2D(1) = BK2D(3)
C
C PRINT THE BESSEL FUNCTIONS OF ORDER L
C
      IF(IDIAG.LE.1)GO TO 190
      WRITE(6,116)
      WRITE(6,105)BJ1A(L+2)
      WRITE(6,105)BK1A(L+2)
      WRITE(6,105)BK1C(L+2)
      WRITE(6,105)BK1D(L+2)
C
      WRITE(6,105)BJ2C(L+2)
      WRITE(6,105)BJ2D(L+2)
      WRITE(6,105)BY2C(L+2)
      WRITE(6,105)BY2D(L+2)
      WRITE(6,105)B12A(L+2)
      WRITE(6,105)B12C(L+2)
      WRITE(6,105)BK2C(L+2)
      WRITE(6,105)BK2D(L+2)
C
C CALCULATE THE 4X4 MATRIX NECESSARY TO EVALUATE THE EXPANSION COEFFICIENTS
C FOR THE OUTER-CORE MODE
C
190 CMAT(1,1) = BJ2C(L+2)
      CMAT(1,2) = BY2C(L+2)
      CMAT(1,3) = -B12C(L+2)
      CMAT(1,4) = 0.0
C
      CMAT(2,1) = AL2*(BJ2C(L+1) - BJ2C(L+3))/2.0
      CMAT(2,2) = AL2*(BY2C(L+1) - BY2C(L+3))/2.0
      CMAT(2,3) = -B12*(B12C(L+1) + B12C(L+3))/2.0
      CMAT(2,4) = 0.0
C
      CMAT(3,1) = BJ2D(L+2)
      CMAT(3,2) = BY2D(L+2)
      CMAT(3,3) = 0.0
      CMAT(3,4) = -BK2D(L+2)
C
      CMAT(4,1) = AL2*(BJ2D(L+1) - BJ2D(L+3))/2.0
      CMAT(4,2) = AL2*(BY2D(L+1) - BY2D(L+3))/2.0
      CMAT(4,3) = 0.0

```



```

C      CMAT(4,4) = B12*(BK2D(L+1) + BK2D(L+3))/2.0
      (
      , , (DIAG.GE.1)WRITE(6,104)
      DO 200 I = 1,4
      IF (DIAG.GE.1)WRITE(6,105)(CMAT(I,J),J=1,4)
      DO 200 J = 1,4
      200 AMAT(I,J) = CMAT(I,J)
C
C      CALL MATINV(CMAT,4,W,DET,0)
C
      IF (DIAG.GE.1)WRITE(6,108)DET
      DO 210 I = 1,3
      W(I) = -AMAT(I,1)
      DO 210 J = 1,3
      210 SINV(I,J) = AMAT(I,J+1)
      CALL MATINV(SINV,3,W,DET,1)
      DO 220 I = 1,3
      220 W(5-I) = W(4-I)
      W(1) = 1.0
C
C      NORMALIZE THE INNER-CORE MODE
C
      I = 1
      CL = -PI*EPS+A*BJ1A(L+3)*BJ1A(L+1)/(2.0*PS1)
      IF (CL.GT.0.0)GO TO 230
      WRITE(6,107)I,CL
      CL = ABS(CL)
      230 CL = 1.0/SQRT(CL)
      GAM = BJ1A(L+2)/BK1A(L+2)
C
C      NORMALIZE THE OUTER-CORE MODE
C
      S1 = B12C(L+2)*2 - B12C(L+1)*B12C(L+3)
      S1 = S1*(W(3)+C)*2/2.0
      XX = BJ2C(L+2)*2 - BJ2D(L+1)*BJ2D(L+3)
      YY = 2.0*BJ2D(L+2)*BY2D(L+2) - BJ2D(L+1)*BY2D(L+3) - BJ2D(L+3)*
      1BY2D(L+1)
      ZZ = BY2D(L+2)*2 - BY2D(L+1)*BY2D(L+3)
      S2 = (XX + W(2)*YY + W(2)*W(2)*ZZ)/2.0
      XX = BJ2C(L+2)*2 - BJ2C(L+1)*BJ2C(L+3)
      YY = 2.0*BJ2C(L+2)*BY2C(L+2) - BJ2C(L+1)*BY2C(L+3) - BJ2C(L+3)*
      1BY2C(L+1)
      ZZ = BY2C(L+2)*2 - BY2C(L+1)*BY2C(L+3)
      S2 = S2 - C*(XX + W(2)*YY + W(2)*W(2)*ZZ)/2.0
      S3 = BK2D(L+2)*2 - BK2D(L+1)*BK2D(L+3)
      S3 = -S3*W(4)*2/2.0
      AL = (S1 + S2 + S3)*PI*EPS
      IF (AL.GT.0.0)GO TO 240
      I = 2
      WRITE(6,107)I,AL
      AL = ABS(AL)
      240 AL = 1.0/SQRT(AL)
C
C      CHECK FOR "PROPER" SIGN OF AL
C
      ARG = AL2*(C+D)/2.0
      CALL BESJ(ARG,ORD,1,ST,NZ)
      XX = ST(1)
      CALL BESYN(ARG,NOR,1,ST)
      ARG = XX + W(2)*ST(1)
      IF (ARG.LT.0.0)AL = -AL
C
C      PRINT THE COEFFICIENTS FOR THE TWO MODES
C
      IF (DIAG.EQ.0)GO TO 251
      ARG = SHINER

```

```

WRITE(6,106)ARG
      = 1.0
      = GAM*CL
WRITE(6,105)ARG,CL
WRITE(6,105)GAM,XX
ARG = SHOUTER
WRITE(6,106)ARG
DO 250 I = 1,4
  XX = AL*W(I)
250 WRITE(6,105)W(I),XX
C
C EVALUATE THE INTEGRALS FOR J1, J2, K1, K2
C
251 J1 = BK1D(L+2)*2 - BK1D(L+1)*BK1D(L+3)
  J1 = J1 - C*C*(BK1C(L+2)*2 - BK1C(L+1)*BK1C(L+3))
  J1 = -EPS*(PI/2.0)*J1*(GAM*CL)*2
C
J2 = B12A(L+2)*2 - B12A(L+1)*B12A(L+3)
J2 = -((A*W(3)*AL)*2)*EPS*J2*PI/2.0
C
K1 = B12*B1A(L+2)*B12A(L+1) - AL1*B1A(L+1)*B12A(L+2)
K1 = -PI*EPS*A*CL*AL*W(3)*K1/(AL1*2 + B12*2)
C
XX = B11*B12D(L+2)*BK1D(L+1) + AL2*B12D(L+1)*BK1D(L+2)
YY = B11*B12C(L+2)*BK1C(L+1) + AL2*B12C(L+1)*BK1C(L+2)
K2 = AL*(XX - C*YY)
XX = B11*B12D(L+2)*BK1D(L+1) + AL2*B12D(L+1)*BK1D(L+2)
YY = B11*B12C(L+2)*BK1C(L+1) + AL2*B12C(L+1)*BK1C(L+2)
K2 = K2 + W(2)*AL*(XX - C*YY)
K2 = GAM*CL*PI*EPS*K2/(AL2*2 + B11*2)
C
C EVALUATE THE INTEGRAL FOR DELTA
C
S1 = -K1
XX = 2.0*B12C(L+2)*BK1C(L+2) + B12C(L+1)*BK1C(L+3) + B12C(L+3)*
  BK1C(L+1)
YY = 2.0*B12A(L+2)*BK1A(L+2) + B12A(L+1)*BK1A(L+3) + B12A(L+3)*
  BK1A(L+1)
S2 = C*C*XX - A*A*YY
S2 = GAM*CL*W(3)*AL*PI*EPS*S2/4.0
S3 = -K2
IF(PS1.NE.PS2)GO TO 350
GO TO 360
350 S4 = B12*B1D(L+2)*BK2D(L+1) - B11*B1D(L+1)*BK2D(L+2)
  S4 = S4/(B11*2 - B12*2)
360 S4 = -GAM*CL*W(4)*AL*PI*EPS*S4
DEL = S1 + S2 + S3 + S4
C
C PRINT THE RESULTS FOR J1, J2, K1, K2, DELTA
C
WRITE(6,109)
XX = THJ
YY = THK
I = 1
J = 2
WRITE(6,110)XX,I,J1
WRITE(6,110)XX,J,J2
WRITE(6,110)YY,I,K1
WRITE(6,110)YY,J,K2
WRITE(6,111)DEL
C
C CALCULATE THE MATRIX ELEMENTS H(I,J):
C
H(1,1) = S1
H(1,2) = S2
H(2,1) = S3
H(2,2) = S4
C
C

```

```

XX = (P1*B)**2
( = (U1 - PS1)*XX
S3 = (K1 - DEL*PS2)*XX
S4 = (K2 - DEL*PS1)*XX
S4 = (J2 - PS2)*XX
WRITE(6,115)
WRITE(6,105)S1,S2
WRITE(6,105)S3,S4
MB = (S2 + S3)/2.0
C
CGAM = (PS1 - PS2)/(K1 + K2)
C
ARG = 4.0*(MB - DEL*S1)*(MB - DEL*S4)
NZ = 0
YY = (S1 - S4)**2
S5 = ABS(ARG)/YY
IF(S5.LT.1.0E-12)NZ = 1
ARG = YY + ARG
ARG = SORT(ARG)
C
E1 = -(S1 + S4 - 2.0*MB*DEL + ARG)/(2.0*XX*(1.0 - DEL*DEL))
E2 = -(S1 + S4 - 2.0*MB*DEL - ARG)/(2.0*XX*(1.0 - DEL*DEL))
IF(E1.LE.E2)GO TO 375
YY = E1
E1 = E2
E2 = YY
375 CONTINUE
C
STORE(JDEX,5) = E1
STORE(JDEX,2) = E2
I = 1
WRITE(6,109)
WRITE(6,112)I,E1
I = 2
WRITE(6,112)I,E2
C
C CALCULATE THE EXPANSION COEFFICIENTS, C AND R, AND CHECK ORTHOGONALITY
C
R1 = -(S1 + XX*E1)/(MB + XX*E1*DEL)
R2 = -(S1 + XX*E2)/(MB + XX*E2*DEL)
C
NZ IS A FLAG INDICATING POSSIBLE ROUNDOFF ERROR IN DETERMINING
R1 OR R2. USE APPROXIMATE FORMULAE.
C
IF(NZ.EQ.0.AND.R2.NE.0.0)GO TO 390
WRITE(6,132)R1,R2
DG = DEL*CGAM
S5 = 2.0*CGAM/(1.0 - DG)
S6 = (1.0 + DG/2.0 + 0.5/DG)
S6 = -DEL*S6/(1.0 + DG)
IF(ABS(R1).GT.1.6)GO TO 385
R1 = S6
R2 = S5
GO TO 390
385 R1 = S5
R2 = S6
C
390 C1 = 1.0 + 2.0*DEL*R1 + R1*R1
C2 = 1.0 + 2.0*DEL*R2 + R2*R2
C1 = 1.0/SORT(C1)
C2 = 1.0/SORT(C2)
WRITE(6,109)
I = 1
WRITE(6,121)I,C1,I,R1
I = 2

```

```

WRITE(6,121)I,C2,I,R2
( = 1.0 + DEL*(R1 + R2) + R1*R2
  ,LTE(6,122)XX
  ARG = SORT(1.0 + CGAM*CGAM)
  XX = CGAM + ARG
  YY = CGAM - ARG
  WRITE(6,125)CGAM,XX,YY
  STORE(JDEX,6) = R1
  SSTORE(JDEX,3) = R2

C CALCULATE THE POWER ASSOCIATED WITH EACH CORE FOR EACH INDIVIDUAL COMPOSIT
C
C INNER-CORE POWER
C
S1 = BU1A(L+2)*0.2 - BU1A(L+1)*BU1A(L+3)
S1 = S1*(A*CL)*0.2
T1 = S1*PI*EPS/2.0
S2 = BT2*BU1A(L+2)*B12A(L+1) - AL1*BU1A(L+1)*B12A(L+2)
S2 = 4.0*CL*W(3)*AL*A*S2/(AL1*0.2 + BT2*0.2)
T2 = S2*PI*EPS/4.0
S3 = S2*R2
S2 = S2*R1
S4 = B12A(L+2)*0.2 - B12A(L+1)*B12A(L+3)
S4 = S4*(W(3)*AL*A)*0.2
T3 = S4*PI*EPS/2.0
S5 = S4*R2*R2
S4 = S4*R1*R1
PC1 = (C1+C1*PI*EPS/2.0)*(S1 + S2 + S4)
PC2 = (C2+C2*PI*EPS/2.0)*(S1 + S3 + S5)
STORE(JDEX,7) = PC1
SSTORE(JDEX,4) = PC2

C OUTER-CORE POWER
C
S1 = BK1D(L+2)*0.2 - BK1D(L+1)*BK1D(L+3)
XX = BK1C(L+2)*0.2 - BK1C(L+1)*BK1C(L+3)
S1 = S1 - C*C*XX
S2 = BU2D(L+2)*0.2 - BU2D(L+1)*BU2D(L+3)
XX = BU2C(L+2)*0.2 - BU2C(L+1)*BU2C(L+3)
S2 = S2 - C*C*XX
S3 = 2.0*BU2D(L+2)*BY2D(L+2) - BU2D(L+1)*BY2D(L+3) -
1 BU2D(L+3)*BY2D(L+1)
XX = 2.0*BU2C(L+2)*BY2C(L+2) - BU2C(L+1)*BY2C(L+3) -
1 BU2C(L+3)*BY2C(L+1)
S3 = S3 - C*C*XX
S4 = BY2D(L+2)*0.2 - BY2D(L+1)*BY2D(L+3)
XX = BY2C(L+2)*0.2 - BY2C(L+1)*BY2C(L+3)
S4 = S4 - C*C*XX
S5 = BT1*BU2D(L+2)*BK1D(L+1) + AL2*BU2D(L+1)*BK1D(L+2)
XX = BT1*BU2C(L+2)*BK1C(L+1) + AL2*BU2C(L+1)*BK1C(L+2)
S5 = S5 - C*C*XX
S6 = BT1*BY2D(L+2)*BK1D(L+1) + AL2*BY2D(L+1)*BK1D(L+2)
XX = BT1*BY2C(L+2)*BK1C(L+1) + AL2*BY2C(L+1)*BK1C(L+2)
S6 = S6 - C*C*XX

SUM = 0.0
XX = C1
YY = R1
I = 0
400 I = I + 1
PR2 = SUM

```

```

11.1.GT.2)GO TO 420
      SUM = SUM
      SUM = S1*(PI*EPS/2.0)*(CL*GAM*XX)**2
      SUM = SUM + S2*(PI*EPS/2.0)*(AL*YY*XX)**2
      SUM = SUM + S3*(PI*EPS*W(2)/2.0)*(AL*YY*XX)**2
      SUM = SUM + S4*(PI*EPS/2.0)*(AL*W(2)*YY*XX)**2
      SUM = SUM - 2.0*S5*XX*XX*PI*EPS*YY*GAM*CL*AL/(AL2**2 + BT1**2)
      SUM = SUM - 2.0*S6*XX*XX*PI*EPS*YY*GAM*CL*AL*W(2)/(AL2**2+BT1**2)
      XX = C2
      YY = R2
      GO TO 400

C
C
C WRITE AND SAVE THE RESULTS OF THE POWER CALCULATION
C
420 STORE(JDEX,8) = PR1
   SSTORE(JDEX,5) = PR2
   XX = (PR1 - PC1)/(PR1 + PC1)
   YY = (PR2 - PC2)/(PR2 + PC2)
   S1 = PR1/PC1
   S2 = PR2/PC2
   STORE(JDEX,9) = S1
   SSTORE(JDEX,6) = S2
   STORE(JDEX,10) = XX
   SSTORE(JDEX,7) = YY
   WRITE(6,123)
   WRITE(6,124)PC1,PR1,S1,XX
   WRITE(6,124)PC2,PR2,S2,YY

C
C
C CALCULATE THE BEAT-LENGTH AND MAXIMUM POWER TRANSFER FOR TWO INTERFERING
C COMPOSITE MODES RESULTING FROM INITIAL CONDITION OF INNER-CORE MODE
C
   BTL = 1.0/(E2 - E1)
   STORE(JDEX,11) = BTL
   WRITE(6,126)BTL
   PT1 = -4.0*(C1**4)*(R1/R2)*(1.0 + DEL*R1)**2
   STORE(JDEX,12) = PT1

C
C
C T1, T2, AND T3 (CALCULATED PREVIOUSLY) ARE THE INTEGRALS OF 1*1, 1*2, AND
C 2*2 OVER THE INNER CORE.
C
   PT2 = T1 + R1*R2*T3 + (R1 + R2)*T2
   PT2 = PT1*PT2
   STORE(JDEX,13) = PT2
   WRITE(6,127)PT1
   WRITE(6,127)PT2

C
C
C CALCULATE THE EXPANSION COEFFICIENTS, D1 AND D2, FOR EXCITATION BY A
C UNIFORM SPOT ON THE AXIS. SPOT SIZES, SPSZ, ARE .05, .10, .15, ... ETC. UP
C
   IF(L.GT.0)GO TO 450
   IF(JDEX.GT.10)GO TO 450
   NOS = -1
   DO 425 I = 1,10
   NOS = NOS + 1
   IF(SPSZ(I).GT.A)GO TO 426
425 CONTINUE
426 WRITE(6,128)
   ORD = 1.0
   DO 430 I = 1,NOS
   AP = SPSZ(I)
   ARG = ALT*AP
   CALL BESJ(ARG,ORD,1,ST,NZ)
   SUM = CL*ST(1)/ALT
   ARG = BT2*AP
   CALL BESJ(ARG,ORD,1,1,ST,NZ)
   ARG = SUM + R1*AL*W(3)*ST(1)/BT2

```

```

S*W = SUM + R2*AL*W(3)*ST(1)/BT2
( = 2.0*SQRT(P1)
L = D2*C1*ARG
D2 = D2*C2*SUN
DRAY(JDEX,1,1) = D1
DRAY(JDEX,2,1) = D2
XX = D1*D1*PC1
YY = D1*D1*PR1
J = 1
WRITE(6,129)AP,J,D1,XX,YY
J = 2
XX = D2*D2*PC2
YY = D2*D2*PR2
430 WRITE(6,129)AP,J,D2,XX,YY
C
C CALCULATE THE FIELDS OF THE TWO UNPERTURBED MODES AND THE TWO SEPARATE
C COMPOSITE MODES.
C
450 ORD = L
WRITE(6,113)
NO = 25
IF((IDTAG.GT.1)NO = 61
DR = 1.20/(NO - 1)
DO 480 I = 1,NO
RH = (I - 1)*DR
IF(RH.GT.A)GO TO 455
ARG = AL1*RH
CALL BESJ(ARG,ORD,1,1,ST,NZ)
F1 = ST(1)*CL
ARG = BT2*RH
CALL BESJ(ARG,ORD,1,1,ST,NZ)
F2 = ST(1)*AL*W(3)
GO TO 470
455 ARG = BT1*RH
CALL BESKN(ARG,L,1,1,ST,NZ)
F1 = CL*GAM*ST(1)
IF(RH.GT.C)GO TO 460
ARG = BT2*RH
CALL BESJ(ARG,ORD,1,1,ST,NZ)
F2 = AL*W(3)*ST(1)
GO TO 470
460 IF(RH.GT.1.0)GO TO 465
ARG = AL2*RH
CALL BESJ(ARG,ORD,1,1,ST,NZ)
F2 = AL*W(1)*ST(1)
CALL BESYN(ARG,L,1,ST)
F2 = F2 + AL*W(2)*ST(1)
GO TO 470
465 ARG = BT2*RH
CALL BESKN(ARG,L,1,1,ST,NZ)
F2 = AL*W(4)*ST(1)
470 CONTINUE
FP = C1*(F1 + R1*F2)
FM = C2*(F1 + R2*F2)
480 WRITE(6,114)RH,F1,F2,FP,FM
GO TO 1
C
900 CONTINUE
IF(JDEX.EQ.0)GO TO 999
WRITE(6,117)
DO 910 I = 1,JDEX
WRITE(6,118)ISS(I),(STORE(I,J),J=1,13)
910 WRITE(6,119)SSSTOR(I,J),J=1,7)
C
C CALCULATE THE TOTAL CORE-TO-RING CROSSTALK, XT, FOR THESE MODES
C

```

147

```

DETER = DETER*PIVOT
330 C(COLUMN,ICOLUMN) = 1.0
340 L = 1,N
350 A(ICOLUMN,L) = A(ICOLUMN,L)/PIVOT
355 IF (N) 380,380,360
360 DO 370 L = 1,M
370 B(ICOLUMN,L) = B(ICOLUMN,L)/PIVOT
C
C REDUCE NON PIVOTS ROWS
C
380 DO 550 L1 = 1,N
390 IF(L1-ICOLUMN) 400,550,400
400 T = A(L1,ICOLUMN)
420 A(L1,ICOLUMN) = 0.
430 DO 450 L = 1,N
450 A(L1,L) = A(L1,L)-A(ICOLUMN,L)*T
455 IF (N) 550,550,460
460 DO 500 L = 1,M
500 B(L1,L) = B(L1,L)-B(ICOLUMN,L)*T
550 CONTINUE
C
C INTERCHANGE COLUMNS
C
600 DO 710 I = 1,N
610 L = N + 1 - I
620 IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
630 JROW = INDEX(L,1)
640 JCOLUMN = INDEX(L,2)
650 DO 705 K = 1,N
660 SWAP = A(K,JROW)
670 A(K,JROW) = A(K,JCOLUMN)
700 A(K,JCOLUMN) = SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
C
750 DETER = 0.0
RETURN
END

```


Program Listing

- BESSELFUNCT-

(BESSELFUNCT routines are used by all of
the preceding programs.)


```

C      IF THIS MESSAGE IS SUPPRESSABLE BY AN ERXSET CALL.
C      THEN EXPLAIN ERXSET USAGE.
      IF (CHARS.GT.0) PRINT 15
15  FORMAT (1/13H ** NOTE **
1/53H TO MAKE THE ERROR MESSAGE PRINTED ABOVE BE NONFATAL.
2/39H OR TO SUPPRESS THE MESSAGE COMPLETELY,
3/37H INSERT AN APPROPRIATE CALL TO ERXSET
4/30H AT THE START OF YOUR PROGRAM.
5/62H FOR EXAMPLE: TO PRINT UP TO 10 NONFATAL WARNING MESSAGES, USE
6/27H      CALL ERXSET(10,0)
      PRINT 20
20  FORMAT (1/28H PROGRAM ABORT DUE TO ERROR.)
      STOP
      END
      SUBROUTINE ONECHK(NCHARS,NARRAY)
C
C      ABSTRACT
C      ONECHK IS A COMPANION ROUTINE OF ERRCHK. IT IS CALLED
C      JUST LIKE ERRCHK, AND MESSAGES FROM IT MAY BE SUPPRESSED
C      BY AN APPROPRIATE CALL TO ERXSET. IT DIFFERS FROM ERRCHK
C      IN THAT EACH CALL TO ONECHK WILL PRODUCE NO MORE THAN ONE
C      PRINTED MESSAGE, REGARDLESS OF HOW MANY TIMES THAT CALL IS
C      EXECUTED, AND ONECHK NEVER TERMINATES EXECUTION.
C      ITS PURPOSE IS TO PROVIDE ONE-TIME-ONLY INFORMATIVE
C      DIAGNOSTICS.
C
C      DESCRIPTION OF ARGUMENTS
C      NCHARS - NUMBER OF CHARACTERS IN THE MESSAGE.
C              IF NEGATED, THE MESSAGE WILL BE PRINTED (ONCE) EVEN
C              IF NFATAL HAS BEEN SET TO 0 (SEE ERXSET).
C      NARRAY - SAME AS IN ERRCHK
C
C      ONECHK USES SUBROUTINES ERRGET, ERRPRT, ERXSET, ERSTGT
C      TO COMPILE DECKS OF ERRCHK
C
C      DIMENSION NARRAY(14)
C      DATA NFLAG/4H,S,*,*/
C      IF (NARRAY(1).EQ.NFLAG) RETURN
C      CALL ERRGET(NF,NT)
C      IF ((NF.EQ.0).AND.(NCHARS.GT.0)) RETURN
C      CALL ERRPRT (59,59H THE FOLLOWING INFORMATIVE DIAGNOSTIC WILL APPEAR
C      ONLY ONCE.)
C      CALL ERRPRT(14BS(NCHARS),NARRAY)
C      IF (NF.GT.0) NF = NF-1
C      CALL ERXSET(NF,NT)
C      NARRAY(1) = NFLAG
C      END
C      SUBROUTINE ERRPRT(NCHARS,NARRAY)
C
C      UTILITY ROUTINE TO SIMPLY PRINT THE HOLLERITH MESSAGE IN NARRAY,
C      WHOSE LENGTH IS NCHARS CHARACTERS.
C
C      DIMENSION NARRAY(14)
C
C      NOTE - NCH MUST BE THE NUMBER OF HOLLERITH CHARACTERS STORED
C      PER WORD. IF NCH IS CHANGED, FORMAT 1 MUST ALSO BE
C      CHANGED CORRESPONDINGLY.
C
C      NCH = 10
C      FOR LINE PRINTERS, USE
C      1  FORMAT (1X,13A10)
C      FOR DATA TERMINALS, USE
C      1  FORMAT (1X,7A10)
C      NWORDS = (NCHARS+NCH-1)/NCH

```

ERCK 660
ERCK 670

ERCK 690
ERCK 700
USEERCK 710
ERCK 720

ERCK 780
ERCK 790
ERCK1350
ERCK1360
ERCK1370
ERCK1380
ERCK1390
ERCK1400
ERCK1410
ERCK1420
ERCK1430
ERCK1440
ERCK1450
ERCK1460
ERCK1470
ERCK1480
ERCK1490
ERCK1510
ERCK1520

ERCK1530

ERCK1640

ERCK 410
ERCK 430
ERCK 440

```

PRINT 1.((NARRAY(I),I=1,NWORDS)
ENCL
SUBROUTINE ERXSET(NFATAL,NTRACE)
ABSTRACT
ERXSET IS A COMPANION ROUTINE TO SUBROUTINE ERRCHK.
ERXSET ASSIGNS THE VALUES OF NFATAL AND NTRACE RESPECTIVELY
TO NF AND NT IN COMMON BLOCK MBLK0 THEREBY SPECIFYING THE
STATE OF THE OPTIONS WHICH CONTROL THE EXECUTION OF ERRCHK.
DESCRIPTION OF ARGUMENTS
BOTH ARGUMENTS ARE INPUT ARGUMENTS OF DATA TYPE INTEGER.
NFATAL - IS A FATAL-ERROR / MESSAGE-LIMIT FLAG. A NEGATIVE
VALUE DENOTES THAT DETECTED DIFFICULTIES ARE TO BE
TREATED AS FATAL ERRORS. NONNEGATIVE MEANS NONFATAL.
A NONNEGATIVE VALUE IS THE MAXIMUM NUMBER OF NONFATAL
WARNING MESSAGES WHICH WILL BE PRINTED BY ERXCHK,
AFTER WHICH NONFATAL MESSAGES WILL NOT BE PRINTED.
(DEFAULT VALUE IS -1.)
NTRACE - .GE.1 WILL CAUSE A TRACE-BACK TO BE GIVEN,
IF THIS FEATURE IS IMPLEMENTED ON THIS SYSTEM.
.LE.0 WILL SUPPRESS ANY TRACE BACK, EXCEPT FOR
CASES WHEN EXECUTION IS TERMINATED.
(DEFAULT VALUE IS 0.)
*NOTE* -- SOME CALLS TO ERXCHK WILL CAUSE UNCONDITIONAL
TERMINATION OF EXECUTION. ERXSET HAS NO EFFECT ON SUCH CALLS.
EXAMPLES
1. TO PRINT UP TO 100 MESSAGES AS NONFATAL WARNINGS USE
CALL ERXSET(100,0)
2. TO SUPPRESS ALL MATHLIB WARNING MESSAGES USE
CALL ERXSET(0,0)
ERXSET USES SUBROUTINES ERSTGT
AND COMPILER DECKS ERRCHK
CALL ERSTGT(0,NFATAL,NTRACE)
RETURN
END
SUBROUTINE ERGET(NFATAL,NTRACE)
ABSTRACT
ERGET IS A COMPANION ROUTINE TO SUBROUTINE ERRCHK.
ERGET ASSIGNS TO FATAL AND NTRACE RESPECTIVELY THE VALUES
OF NF AND NT IN COMMON BLOCK MBLK0 THEREBY ASCERTAINING THE
STATE OF THE OPTIONS WHICH CONTROL THE EXECUTION OF ERRCHK.
DESCRIPTION OF ARGUMENTS
BOTH ARGUMENTS ARE OUTPUT ARGUMENTS OF DATA TYPE INTEGER.
NFATAL - CURRENT VALUE OF NF (SEE DESCRIPTION OF ERXSET.)
NTRACE - CURRENT VALUE OF NT (SEE DESCRIPTION OF ERXSET.)
CALL ERSTGT(1,NFATAL,NTRACE)
RETURN
END
SUBROUTINE ERSTGT(K,NFATAL,NTRACE)
THIS ROUTINE IS A SLAVE TO ERGET AND ERSET WHICH KEEPS
THE FLAGS AS LOCAL VARIABLES.
*** IF LOCAL VARIABLES ARE NOT NORMALLY RETAINED BETWEEN
CALLS ON THIS SYSTEM, THE VARIABLES LNF AND LNT CAN BE

```

PLACED IN A COMMON BLOCK AND PRESET TO THE FOLLOWING
VARY 5 IN THE MAIN PROGRAM.

```
DATA LNF/-1/.LNT/0/
IF (K.LE.0) LNF = NFATAL
IF (K.LE.0) LNT = NTRACE
IF (K.GT.0) NFATAL = LNF
IF (K.GT.0) NTRACE = LNT
RETURN
END
FUNCTION BESK01(X,NU,KODE,NZ)
```

WRITTEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974.

REFERENCE SAND-75-0149

ABSTRACT

BESK01 COMPUTES BESSEL FUNCTIONS $K/\text{SUB}(NU)/(X)$, $NU=0$ OR 1
OR SCALED BESSEL FUNCTIONS $\text{EXP}(X)*K/\text{SUB}(NU)/(X)$,
 $NU=0$ OR 1 FOR $X.GT.0$. CHEBYSHEV EXPANSIONS, PROPERLY SCALED
FOR SMALL AND LARGE X , ARE USED ON INTERVALS $0.LT.X.LE.2$,
 $2.LT.X.LE.5$, AND $X.GT.5$. THE UNDERFLOW TEST IS MADE ON
 $X.LE.ELIM$ WITH $ELIM=667$. BESK01 CALLS FUNCTION BES101.

DESCRIPTION OF ARGUMENTS

INPUT

X - $0.LT.X.LE.667$. FOR $KODE=1$, $X.GT.0$ FOR $KODE=2$
NU - ORDER DESIRED, $NU=0$ OR 1
KODE - A PARAMETER TO INDICATE THE SCALING OPTION
KODE=1 RETURNS $\text{ANS}=K/\text{SUB}(NU)/(X)$, $NU=0$ OR 1
KODE=2 RETURNS $\text{ANS}=\text{EXP}(X)*K/\text{SUB}(NU)/(X)$, $NU=0$ OR 1

OUTPUT

BESK01 - K BESSEL FUNCTION OF ORDER NU AT X SCALED ACCORDING
TO KODE
NZ - UNDERFLOW INDICATOR
NZ=0, NORMAL RETURN, COMPUTATION COMPLETED
NZ.NE.0, ANS SET TO ZERO DUE TO UNDERFLOW WITH
KODE=1 AND $X.GT.667$

ERROR CONDITIONS

IMPROPER INPUT ARGUMENTS - A FATAL ERROR
UNDERFLOW WITH $KODE=1$ - A NON-FATAL ERROR(NZ.NE.0)

BESK01 USES SUBROUTINES BES101, ERRCHK, ERRGET, ERRPRT,
ERRSET, ERSTGT
COMPILE DECKS BESK01, BES101, ERRCHK

DIMENSION AKO(15),BKO(21),CKO(14),AK1(15),BK1(22),CK1(14)

DATA ELIM / 667. /

DATA N1,N2,N3,N4/15,21,14,22/

DATA M1,M2,M3,M4/13,19,12,20/

DATA EMLT/-1.15931515658412E-1/

DATA AKO

```
1 2.20263147944787E-01, 2.94143920648988E-02, 4.78389179259074E-03,
2 3.65095673281410E-04, 3.96552105234993E-05, 2.13571244350405E-06,
3 1.74123741003729E-07, 7.26649791536202E-09, 4.74231032016171E-10,
4 1.61749721604899E-11, 8.8011420783973E-13, 2.53991582114493E-14,
5 1.18507018244790E-15/
DATA BKO / 1.21101810253138E+00, 1.66889519207676E-02,
1-3.32975981412374E-03, 6.70802900717429E-04, -1.36272865384502E-04,
2 2.78660969643180E-05, -5.74300240638451E-06, 1.18942254775776E-06,
3 -2.47573075834394E-07, 5.17613906627857E-08, -1.0865350852187E-08,
4 2.28899509247299E-09, -4.83794289746110E-10, 1.02556286108303E-10,
```

```

5-2.17950101893024E-11, 4.64499519650372E-12, -9.92020238192219E-13,
6 2.30748247184E-13, -4.55250947249326E-14, 9.77948196627E-15,
7-2.428853630363E-15/
DATA CKO / 1.23878593820170E+00, -1.41756153741679E-02,
1 3.37809034043106E-04, -1.35201655758935E-05, 7.92815326397551E-07,
2-5.64584651140153E-08, 4.75381608069133E-09, -4.56892133422380E-10,
3 4.89340910679015E-11, -5.74009881252931E-12, 7.27938173682122E-13,
4-9.88017112659449E-14, 1.32381749918418E-14, -2.16443591571975E-15/
DATA AK1 / 4.62486240842478E-01, 6.05181915156673E-01,
1 1.53253828510709E-01, 3.35117014379959E-02, 3.36879962666501E-03,
2 4.41042696291495E-04, 2.83189625853098E-05, 2.64548793222978E-06,
3 1.25637058646502E-07, 9.12550891350504E-09, 3.44783302431681E-10,
4 2.04890221867815E-11, 6.43563932230815E-13, 3.23636474094122E-14,
5 8.70458673285249E-16/
DATA BK1 / 1.38930840300441E+00, -5.89753920597669E-02,
1 1.20249909184072E-02, -2.55387533195591E-03, 5.45258387639594E-04,
2-1.16936329705121E-04, 2.51746904352026E-05, -5.43777151950107E-06,
3 1.17797109704500E-06, -2.55830051061306E-07, 5.568564833337E-08,
4-1.21451129231835E-08, 2.65359402057287E-09, -5.80715571337876E-10,
5 1.27268495790715E-10, -2.79285412641216E-11, 6.13612949711377E-12,
6-1.34963109251970E-12, 2.97146532041432E-13, -6.54827012807628E-14,
7 1.44428209792469E-14, -3.18800891899413E-15/
DATA CK1 / 1.29837185686353E+00, 4.4449150892785E-02,
1-5.87146375780839E-04, 2.02849692864934E-05, -1.05926607637730E-06,
2 7.16046166444819E-08, -5.82233926413546E-09, 5.45714706062071E-10,
3-5.73482672655767E-11, 6.62769161077494E-12, -8.30462152737046E-13,
4 1.11605017034540E-13, -1.59497634434667E-14, 2.40744099380385E-15/
C
BESK01 = -X*IND
IF (MODE.LT.1.OR.MODE.GT.2) GO TO 90
IF (NU.LT.0.OR.NU.GT.1) GO TO 91
IF (X.LE.0.) GO TO 92
IK=NU+1
NZ=0
GO TO (100,200),IK
C
K/SUB(0)/(X) BESSEL FUNCTION
C
100 CONTINUE
IF (X.GT.2.) GO TO 110
T=X-1.
TT=T+T
J=N1
F1=AK0(J)
F2=0.
DO 105 I=1,M1
J=J-1
TEMP1=F1
F1=TT-F1-F2*AK0(J)
F2=TEMP1
105 CONTINUE
BNS=BESJ01(X,0,1)
ANS=-(EMLT+ALOG(X))*BNS*(T+F1-F2*AK0(1))
GO TO (106,107), MODE
107 BESK01=ANS*EXP(X)
RETURN
108 BESK01=ANS
RETURN
110 IF (X.GT.5.) GO TO 120
T=(X+X-7.)/3.
TT=T+T
J=N2
F1=BK0(J)
F2=0.
DO 115 I=1,M2
J=J-1

```

```

TEMP=F1
F1=(F1-F2+BK0(J)
F2=TEMP1
115 CONTINUE
ANS=(T*F1-F2+BK0(1))/SORT(X)
GO TO (116,106),KODE
116 BESK01=ANS*EXP(-X)
RETURN
120 CONTINUE
T=10./X-1.
TT=T+T
J=N3
F1=CK0(J)
F2=0.
DO 125 I=1,M3
J=J-1
TEMP1=F1
F1=TT*F1-F2+CK0(J)
F2=TEMP1
125 CONTINUE
ANS=(T*F1-F2+CK0(1))/SORT(X)
GO TO (126,106),KODE
126 IF(X.GT.ELIM) GO TO 93
GO TO 116
C
C K/SUB(1)/(X) BESSEL FUNCTION
C
200 CONTINUE
IF(X.GT.2.) GO TO 210
T=X-1.
TT=T+T
J=N1
F1=AK1(J)
F2=0.
DO 205 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2+AK1(J)
F2=TEMP1
205 CONTINUE
BNS=BESIO1(X,1,1)
ANS=((EMLT+ALOG(X))*BNS+1./X-(T*F1-F2+AK1(1)))
GO TO (106,107),KODE
210 IF(X.GT.5.) GO TO 220
T=(X+X-7.)/3.
TT=T+T
J=N4
F1=BK1(J)
F2=0.
DO 215 I=1,M4
J=J-1
TEMP1=F1
F1=TT*F1-F2+BK1(J)
F2=TEMP1
215 CONTINUE
ANS=(T*F1-F2+BK1(1))/SORT(X)
GO TO (116,106),KODE
220 CONTINUE
T=10./X-1.
TT=T+T
J=N3
F1=CK1(J)
F2=0.
DO 225 I=1,M3
J=J-1
TEMP1=F1

```

```

      F1=F1-F2+CK1(J)
      F2= F1
225 CONTINUE
      ANS=(T+F1-F2+CK1(1))/SORT(X)
      GO TO (226,106) ,MODE
226 IF(X.GT.ELIM) GO TO 93
      GO TO 116
90 CONTINUE
      CALL ERRCHK(44,44HIN BESK01, SCALING OPTION, MODE, NOT 1 OR 2.)
      RETURN
91 CONTINUE
      CALL ERRCHK(33,33HIN BESK01, ORDER, NU, NOT 0 OR 1.)
      RETURN
92 CONTINUE
      CALL ERRCHK(40,40HIN BESK01, X LESS THAN OR EQUAL TO ZERO.)
      RETURN
93 CONTINUE
      BESK01=0.
      NZ=1
      RETURN
      END
      FUNCTION BESY01(X,NU,ANSJ)

```

WRITTEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974

REFERENCE SAND-75-0148

ABSTRACT

BESY01 COMPUTES BESSEL FUNCTIONS $J(\text{SUB}(NU))/(X)$ AND $Y(\text{SUB}(NU))/(X)$, $NU=0$ OR 1 FOR $X.GT.0$. RATIONAL CHEBYSHEV APPROXIMATIONS, ASYMPTOTICALLY SCALED FOR SMALL AND LARGE X , ARE USED ON INTERVALS $0.LE.X.LE.8$ AND $X.GT.8$. THE COST IN RETURNING $J(\text{SUB}(NU))/(X)$ IS MINIMAL SINCE THIS FUNCTION IS NEEDED IN THE ASYMPTOTIC FORM FOR $0.LT.X.LE.8$ AND ONLY REQUIRES A REARRANGEMENT OF 4 FACTORS NEEDED FOR $X.GT.8$. BESY01 CALLS FUNCTION BESJ01.

DESCRIPTION OF ARGUMENTS

INPUT

X - X.GT.0.
NU - ORDER DESIRED, $NU=0$ OR 1

OUTPUT

BESY01 - Y BESSEL FUNCTION OF ORDER NU AT X
ANSJ - J BESSEL FUNCTION OF ORDER NU AT X

ERROR CONDITIONS

IMPROPER INPUT ARGUMENTS - A FATAL ERROR

BESY01 USES SUBROUTINES BESJ01, ERRCHK, ERRGET, ERRPRT,

ERRSET, ERSTGT

COMPILE DECKS BESY01, BESJ01, ERRCHK

DIMENSION P0(8),Q0(8),P1(7),Q1(8),P0N(4),P0D(5),Q0N(4),Q0D(5),

1 P1N(4),P1D(5),Q1N(4),Q1D(5)

DATA N1,N2,N3,N4,M1,M2,M3,M4/B,7.5,4,6,5,3,2/

DATA P104,TFPI,TOPI / 7.85398163397448E-01, 2.35619449019234E+00,
1 6.36619772367581E-01/

DATA P0 /-1.01860971224170E+16, 2.43804485256950E+16,
1-1.90205266308270E+15, 4.77091764845963E+13,-5.28348803896724E+11,
2 2.86910828729839E+09,-7.53147545781676E+06, 7.70593364872924E+03/


```

DAT. 00 / 1.38014963863775E+17, 1.78633877994688E+15,
1 752268723807E+13, 5.21829549438144E+10, 1.71651744157E+08,
2 4.467794501471E+05, 8.18358066417824E+02, 1.000000000000E+00,
DATA P1 /-1.06525580073146E+16, 2.80179513474828E+15,
1-1.19784970206113E+14, 1.91351887731600E+12, -1.38006319639808E+10,
2 4.57445909094871E+07, -5.70683926136436E+04/
DATA Q1 / 5.43339594207058E+16, 7.71123767247204E+14,
1 5.61619655823693E+12, 2.77405419245635E+10, 1.02955286238916E+08,
2 2.97764175120939E+05, 8.58392393694713E+02, 1.000000000000E+00/
DATA P0N / 5.43387249648474E+05, 3.99030011721535E+05,
1 6.08286198525507E+04, 1.57972732370604E+03/
DATA P0D / 6.81034922021421E+05, 5.70858162215322E+05,
1 7.67689810942388E+04, 2.05200863309931E+03, 1.000000000000E+00/
DATA P1N /-4.21538738930469E+05, -2.97874854285400E+05,
1-4.26857048542501E+04, -9.89621122975118E+02/
DATA P1D /-5.28320460927703E+05, -3.72363382370422E+05,
1-5.28353786226566E+04, -1.15530681849500E+03, 1.000000000000E+00/
DATA Q0N /-4.92207223828469E+02, -4.39316681986953E+02,
1-8.58904583816188E+01, -3.12667145228239E+00/
DATA Q0D / 3.94809774152373E+04, 3.55999748740455E+04,
1 7.19786937649468E+03, 3.02885800333886E+02, 1.000000000000E+00/
DATA Q1N / 3.68932567438047E+03, 3.37134265139743E+03,
1 6.32384402139958E+02, 2.17125499052971E+01/
DATA Q1D / 1.03990279374871E+05, 9.05851244162508E+04,
1 1.72767909550058E+04, 6.40089571203976E+02, 1.000000000000E+00/

```

```

C BESY01 = -X*IND
IF(NU.LT.0.OR.NU.GT.1) GO TO 91
IF(X.LE.0.) GO TO 92
JS=NU+1
IF(X.GT.8.) GO TO 200
X2=X*X
X=LE.X
GO TO (10,20),JS

```

```

C Y/SUB(0)/(X) AND J/SUB(0)/(X) BESSEL FUNCTION
10 CONTINUE
SUMP=P0(N1)*X2+P0(N2)
SUMQ=Q0(N1)*X2+Q0(N2)
J=M1
DO 15 I=1,M1
SUMP=SUMP+X2*P0(J)
SUMQ=SUMQ+X2*Q0(J)
J=J+1
15 CONTINUE
ANS=SUMP/SUMQ
ANSJ=BESJ01(X,0)
BESY01=TOPI*ALOG(X)*ANSJ+ANS
RETURN

```

157

```

C Y/SUB(1)/(X) AND J/SUB(1)/(X) BESSEL FUNCTION
20 CONTINUE
SUMP=P1(N2)*X2+P1(M1)
SUMQ=Q1(N1)*X2+Q1(N2)
J=M1
DO 25 I=1,M2
SUMQ=SUMQ+X2*Q1(J)
J=J+1
SUMP=SUMP+X2*P1(J)
25 CONTINUE
SUMQ=SUMQ+X2*Q1(J)
ANS=X*(SUMP/SUMQ)
ANSJ=BESJ01(X,1)
BESY01=TOPI*(ALOG(X)*ANSJ-1./X)+ANS
RETURN

```

```

C      X,GT,.8
C 200 CD/ NUC
      Z=b,X
      Z2=Z*Z
      GO TO (30,40),JS
C      Y/SUB(0)/(X) AND J/SUB(0)/(X) BESSEL FUNCTION
C
C 30 CONTINUE
      SUMPN=PON(N4)*Z2+PON(M3)
      SUMQN=QON(N4)*Z2+QON(M3)
      SUMPD=POD(M2)*Z2+POD(N4)
      SUMQD=QOD(M2)*Z2+QOD(N4)
      J=M3
      DO 35 I=1,M4
      SUMPD=SUMPD+Z2+POD(J)
      SUMQD=SUMQD+Z2+QOD(J)
      J=J+1
      SUMPN=SUMPN+Z2+PON(J)
      SUMQN=SUMQN+Z2+QON(J)
C 35 CONTINUE
      SUMPD=SUMPD+Z2+POD(J)
      SUMQD=SUMQD+Z2+QOD(J)
      XV=X-PI/4
      GO TO 50
C      Y/SUB(1)/(X) AND J/SUB(1)/(X) BESSEL FUNCTION
C
C 40 CONTINUE
      SUMPN=PIN(N4)*Z2+PIN(M3)
      SUMQN=QIN(N4)*Z2+QIN(M3)
      SUMPD=PID(M2)*Z2+PID(N4)
      SUMQD=QID(M2)*Z2+QID(N4)
      J=M3
      DO 45 I=1,M4
      SUMPD=SUMPD+Z2+PID(J)
      SUMQD=SUMQD+Z2+QID(J)
      J=J+1
      SUMPN=SUMPN+Z2+PIN(J)
      SUMQN=SUMQN+Z2+QIN(J)
C 45 CONTINUE
      SUMPD=SUMPD+Z2+PID(J)
      SUMQD=SUMQD+Z2+QID(J)
      XV=X-1/2*PI
C 50 CONTINUE
      CXV=COS(XV)
      SXV=SIN(XV)
      PV=SUMPN/SUMPD
      QV=(SUMQN/SUMQD)*Z
      RRTX=1./SQRT(X)
      ANSJ=(PV*CXV-OV*SV)*RRTX
      BESY01=(QV*CXV+PV*SV)*RRTX
      RETURN
C 91 CONTINUE
      CALL ERRCBK(33,33HIN BESY01, ORDER, NU, NOT 0 OR 1.)
      RETURN
C 92 CONTINUE
      CALL ERRCBK(40,40HIN BESY01, X LESS THAN OR EQUAL TO ZERO.)
      RETURN
      END
      FUNCTION BESJ01(X,NU)
C
C WRITTEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974
C
C REFERENCE SAND-75-0148
C

```

```

C C      , JSU01 COMPUTES BESSEL FUNCTIONS J/SUBNU/(X), NU=0 OR
C C      .OR REAL, UNRESTRICTED X, RATIONAL CHEBYSHEV APPROXIMATE .AS,
C C      ASYMPTOTICALLY SCALED FOR SMALL AND LARGE ABS(X). ARE USED ON
C C      O.L.E.X.LE.B. AND X.G.T.B. THE SIGN IS FIXED ACCORDING TO THE
C C      EVENNESS OR ODONESS OF THE FUNCTION.
C C
C C DESCRIPTION OF ARGUMENTS
C C
C C INPUT          - UNRESTRICTED
C C              X   - ORDER DESIRED, NU=0 OR 1
C C
C C OUTPUT         BSJ01 - J BESSEL FUNCTION OF ORDER NU AT X
C C
C C ERROR CONDITIONS
C C IMPROPER INPUT ARGUMENTS - A FATAL ERROR
C C
C C BSJ01 USES SUBROUTINES ERRCHK, ERGETT, ERKSET, ERSTGT
C C COMPILE DECKS BJSJ01, ERRCHK
C C
C C DIMENSION PO(B),QO(8),P(17),Q(18),PON(4),POD(S),QON(4),QOD(S),
C C     P1N(4),PID(S),QIN(4),QID(S)
C C
C C DATA N1,N2,N3,N4,M1,M2,M3,M4/B,7.5,4.6,5.3,2./
C C DATA PI04,TFPI / 7.85398163397448E-01, 2.35619449019234E+00,/
C C
C C DATA PO / 1.-85962317621898E+17,-4.41458293918160E+16,
C C     1 2.33448917187878E+15,-4.77855594426736E+13, 4.62172225031719E+11,
C C     2-2.27149043955360E+09, 5.51358456477075E+06,-5.29261713038456E+03,/
C C DATA QO / 1.85962317621898E+17, 2.34475001365900E+15,
C C     1 1.50154624497698E+13, 6.4398674535133E+10, 2.04251483521344E+08,
C C     2 4.94030794918140E+05, 8.84230367561755E+02, 1.00000000000000E+00,/
C C DATA P1 / 3.765461264493808E+16,-4.187039939363345E+15,
C C     1 1.34815539792856E+14,-1.81993375043619E+12, 1.16648876892748E+10,
C C     2 3.5357426649922E+07, 4.10201396310345E+04,/
C C DATA Q1 / 7.53092248987613E+16, 1.03957332507946E+15,
C C     1 7.34219887157498E+12, 3.50368394634232E+10, 1.25012552311935E+08,
C C     2 3.45178964292159E+05, 7.20983988573980E+02, 1.00000000000000E+00,/
C C DATA POD / 5.43387249648474E+05, 3.99030011721535E+05,
C C     1 6.0826619855507E+04, 1.57927232370604E+03,
C C     2 DATA PID / 6.81034222021421E+05, 5.00858162215322E+05,
C C     1 7.67699810942388E+04, 2.0520083309931E+03, 1.00000000000000E+00,/
C C DATA PIN / -4.21538738930469E+05,-2.97874854285400E+05,
C C     1-4.26857048542501E+04,-9.89521122975118E+02,
C C     2 DATA PID / -5.28320460927703E+05,-3.72363382370422E+05,
C C     1-5.28353786226566E+04,-1.15530681849500E+03, 1.00000000000000E+00,/
C C     2 DATA PON / -4.92207223828496E+02,-4.39316681986953E+02,
C C     1-8.58904593816188E+01,-3.128607145228239E+00,
C C     2 DATA QOD / 3.94809774152373E+04, 3.55999748704455E+04,
C C     1 7.19786997649458E+03, 3.32985000333866E+02, 1.00000000000000E+00,/
C C     2 DATA QIN / 3.88932367438047E+03, 3.37134265139743E+03,
C C     1 6.323944402139958E+02, 2.17125499052971E+01,
C C     2 DATA QID / 1.03990379374871E+05, 9.05851244162508E+04,
C C     1 1.12787909550058E+04, 6.40089571203976E+02, 1.00000000000000E+00,/
C C
C C BSJ01 -- KIND
C C IF(NU.LT.O.OR.NU.GT.1) GO TO 91
C C AX=ABS(X)
C C JS=NU+1
C C IF(AH.GT.B.) GO TO 200
C C X2=X*X
C C XLE.B

```

```

GO TO (10,20),JS
C
C J/SUB(0)/(X) BESSEL FUNCTION
C
10 CONTINUE
SUMP=P0(N1)*X2+P0(N2)
SUMQ=Q0(N1)*X2+Q0(N2)
J=M1
DO 15 I=1,M1
SUMP=SUMP*X2+P0(J)
SUMQ=SUMQ*X2+Q0(J)
J=J+1
15 CONTINUE
BESJ01=SUMP/SUMQ
RETURN
C
C J/SUB(1)/(X) BESSEL FUNCTION
C
20 CONTINUE
SUMP=P1(N2)*X2+P1(M1)
SUMQ=Q1(N1)*X2+Q1(N2)
J=M1
DO 25 I=1,M2
SUMP=SUMP*X2+P1(J)
SUMQ=SUMQ*X2+Q1(J)
J=J+1
25 CONTINUE
SUMP=SUMP*X2+P1(J)
SUMQ=SUMQ*X2+Q1(J)
BESJ01=X*(SUMP/SUMQ)
RETURN
C
C X.GT..8
200 CONTINUE
Z=8./AX
Z2=Z*Z
GO TO (30,40),JS
C
C J/SUB(0)/(X) BESSEL FUNCTION
C
30 CONTINUE
SUMP=P0(N4)*Z2+P0(N3)
SUMQ=Q0(N4)*Z2+Q0(N3)
SUMP=P0(N2)*Z2+P0(N4)
SUMQ=Q0(N2)*Z2+Q0(N4)
J=M3
DO 35 I=1,M4
SUMP=SUMP*Z2+P0(J)
SUMQ=SUMQ*Z2+Q0(J)
J=J+1
SUMP=SUMP*Z2+P0(J)
SUMQ=SUMQ*Z2+Q0(J)
35 CONTINUE
SUMP=SUMP*Z2+P0(J)
SUMQ=SUMQ*Z2+Q0(J)
XV=AX-PI04
BESJ01=((SUMP/SUMP)*COS(XV)-Z*(SUMQ/SUMQ)*SIN(XV))/SQRT(AX)
RETURN
C
C J/SUB(1)/(X) BESSEL FUNCTION
C
40 CONTINUE
SUMP=P1(N4)*Z2+P1(N3)
SUMQ=Q1(N4)*Z2+Q1(N3)
SUMP=P1(N2)*Z2+P1(N4)
SUMQ=Q1(N2)*Z2+Q1(N4)
J=M3
DO 45 I=1,M4

```

```

SUMPO=SUMPO+Z2*PID(J)
SU( J )=SUMQD+Z2*Q1D(J)
J=J+1
SUMPN=SUMPN+Z2*PIN(J)
SUMON=SUMON+Z2*QIN(J)
45 CONTINUE
SUMPD=SUMPD+Z2*PID(J)
SUMQD=SUMQD+Z2*Q1D(J)
XV=AX-TFPI
ANS=((SUMPN/SUMPD)*COS(XV)-Z*(SUMON/SUMQD)*SIN(XV))/SQRT(AX)
BESJ01=ANS*SIGN(1...X)
RETURN
91 CONTINUE
CALL ERRCHK(33.33HIN BESJ01, ORDER, NU, NOT 0 OR 1.)
RETURN
END
FUNCTION BESJ01(X,NU,MODE)
WRITTEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974.
REFERENCE SAND-75-0149
ABSTRACT
BESJ01 COMPUTES BESSEL FUNCTIONS  $I_{\nu}(X)$  AND  $J_{\nu}(X)$  FOR  $\nu=0$  OR  $1$ 
OR SCALED BESSEL FUNCTIONS  $\exp(-\text{ABS}(X)) * I_{\nu}(\text{SUB}(NU)/(X))$  AND  $J_{\nu}(\text{SUB}(NU)/(X))$ 
FOR SMALL AND LARGE  $\text{ABS}(X)$ . ARE USED ON INTERVALS  $0 \leq X \leq 4$ ,
 $4 \leq X \leq 8$ , AND  $X \geq 8$ . THE SIGN IS FIXED ACCORDING TO THE
EVENNESS OR ODDNESS OF THE FUNCTION. THE OVERFLOW TEST IS MADE
ON  $\text{ABS}(X) \cdot \text{LE} \cdot \text{ELIM}$  WITH  $\text{ELIM}=667$ .
DESCRIPTION OF ARGUMENTS
INPUT
X -  $\text{ABS}(X) \cdot \text{LE} \cdot 667$ . FOR  $\text{MODE}=1$ , UNRESTRICTED FOR  $\text{MODE}=2$ 
NU - ORDER DESIRED,  $\text{NU}=0$  OR  $1$ 
MODE - A PARAMETER TO INDICATE THE SCALING OPTION
       $\text{MODE}=1$  RETURNS  $\text{ANS} = I_{\nu}(\text{SUB}(NU)/(X))$ ,  $\text{NU}=0$  OR  $1$ 
       $\text{MODE}=2$  RETURNS  $\text{ANS} = \exp(-X) * I_{\nu}(\text{SUB}(NU)/(X))$ ,  $\text{NU}=0$  OR  $1$ 
OUTPUT
BESJ01 -  $I_{\nu}$  BESSEL FUNCTION OF ORDER  $\text{NU}$  AT  $X$  SCALED ACCORDING
      TO  $\text{MODE}$ 
ERROR CONDITIONS
IMPROPER INPUT ARGUMENTS - A FATAL ERROR
OVERFLOW FOR  $\text{MODE}=1$  - A FATAL ERROR
BESJ01 USES SUBROUTINES ERRCHK, ERRGET, ERRPRT, ERRSET, ERSTGT
COMPILER DECKS BESJ01, ERRCHK
DIMENSION A10(18),B10(18),C10(19),A11(18),B11(18),C11(19)
DATA ELIM / 667. /
DATA M1,M2/18,19/
DATA M1,M2/16,17/
DATA A10
1 1.93239151962331E+00, 5.12072928951984E-01, 1.19244431087252E-01,
2 2.04767667749564E-02, 3.29580433784138E-03, 4.17924020345661E-04,
3 5.1365890926708E-05, 5.16036527683877E-06, 5.12886756184602E-07,
4 4.26546791785515E-08, 3.5582325462485E-09, 2.52433036418873E-10,
5 1.81419096716532E-11, 1.12207674472741E-12, 7.08316386761487E-14,
6 3.88308114508344E-15/
DATA B10
1 4.27652251957361E-03, -8.56652186435765E-04, 1.08431306444222E-04,

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

2-1 1385649051770E-05, 3.18457963070919E-06, -5.48115448889 E-07,
3 4.22416818253E-08, -1.5082504887134E-08, 2.35026532228 E-09,
4-3.4904194258885E-10, 4.91824029507119E-11, -6.5670859174462E-12,
5 8.3093935382694E-13, -9.97106489731714E-14, 1.13614195790750E-14,
6-1.2311094739334E-15/
DATA C10 / 4.02245205507054E-01, 3.36911647825569E-03,
1 6.88975834691682E-05, 2.89137052083476E-06, 2.04891858946905E-07,
2 2.2666889049816E-08, 3.39623202570920E-09, 4.94060238821134E-10,
3 1.18891471071382E-11, -3.14991652807488E-11, -1.32158118398615E-11,
4-1.79417853277908E-12, 7.18012446526379E-13, 3.85277837074653E-13,
5 1.54008635387642E-14, 4.15056915510500E-14, -9.55484713964049E-15,
6 3.81168204982708E-15, 1.77256039316563E-15/
DATA A11 / 1.11729135052299E+00, 8.95198877808018E-01,
1 3.36971231909350E-01, 7.24336830339257E-02, 1.53300913475520E-02,
2 2.26237207566113E-03, 3.36089476871761E-04, 3.78329005043256E-05,
3 4.3436708173434E-06, 3.95545350342659E-07, 3.70599583606154E-08,
4 2.83473074207374E-09, 2.24494698516683E-10, 1.4806707975197E-11,
5 1.01588206538587E-12, 5.89015292859435E-14, 3.56565058329046E-15,
6 1.84456468936977E-16/
DATA B11 / 1.5423684772347E-01, -2.20987741718608E-02,
1 2.19839683870564E-03, -2.0705598880429E-04, 1.22621848833534E-05,
2 1.43239687327040E-06, -7.91717404214453E-07, 2.15368543162913E-07,
3-4.70664774819712E-08, 9.04215441510569E-09, -1.57608499172581E-09,
4 2.52964136818147E-10, -3.77065425810734E-11, 5.25015808110040E-12,
5-6.85889299579377E-13, 8.43830741608493E-14, -9.80745080362827E-15,
6 1.07990041545005E-15/
DATA C11 / 3.89288117509140E-01, -9.76109749136147E-03,
1-1.10588938762624E-04, -3.88256480887769E-06, -2.51233623787023E-07,
2-2.6314688468956E-08, -3.8353803858299E-09, -5.58974346220626E-10,
3-1.8974958125807E-11, 3.25260358287477E-11, 1.41258074355730E-11,
4 2.03562854596022E-12, -7.19855176877244E-13, -4.08355112068096E-13,
5-2.10154176162218E-14, 4.27244003763443E-14, 1.04202768179381E-14,
6-3.81440283617941E-15, -1.88035459733009E-15/
C
BES101 = -X*IND
IF (MODE, LT, 1, OR, CODE, GT, 2) GO TO 90
IF (NU, LT, 0, OR, NU, GT, 1) GO TO 91
AX=ABS(X)
IX=NU+1
GO TO (100,200),IX
C
I/SUB(O)/(X) BESSEL FUNCTION
C
100 CONTINUE
IF (AX, GT, 4.) GO TO 110
TT=AX-2.
T=TT*.5
J=N1
F1=A10(J)
F2=0.
DO 105 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2+A10(J)
F2=TEMP1
105 CONTINUE
ANS=T*F1-F2+A10(1)
GO TO (106,107), CODE
107 BES101=ANS*EXP(-AX)
RETURN
108 BES101=ANS
RETURN
110 IF (AX, GT, 8.) GO TO 120
TT=AX-6.
T=TT*.5
J=N1

```

```

F1=1.0(U)
F2
DO 115 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2+B10(U)
F2=TEMP1
115 CONTINUE
ANS=T*F1-F2+B10(1)
GO TO (116,108),MODE
116 BES101=ANS*EXP(AX)
120 RETURN
120 CONTINUE
T=16./AX-1.
TT=T*T
J=N2
F1=C10(U)
F2=0.
DO 125 I=1,M2
J=J-1
TEMP1=F1
F1=TT*F1-F2+C10(U)
F2=TEMP1
125 CONTINUE
ANS=(T*F1-F2+C10(1))/SORT(AX)
GO TO (126,106),MODE
126 IF(AX.GT.ELIM) GO TO 92
BES101=ANS*EXP(AX)
RETURN
C
C 1/SUB(1)/(X) BESSEL FUNCTION
C
200 CONTINUE
IF(AX.GT.4.) GO TO 210
TT=AX-2.
T=TT*.5
J=N1
F1=A11(U)
F2=0.
DO 205 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2+A11(U)
F2=TEMP1
205 CONTINUE
ANS=X*{(T*F1-F2+A11(1))
GO TO (106,107),MODE
210 IF(AX.GT.8.) GO TO 220
TT=AX-6.
T=TT*.5
J=N1
F1=B11(U)
F2=0.
DO 215 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2+B11(U)
F2=TEMP1
215 CONTINUE
ANS=T*F1-F2+B11(1)
GO TO (216,217), MODE
216 ANS=ANS*EXP(AX)
217 BES101=SIGN(ANS,X)
RETURN
220 CONTINUE
T=16./AX-1.

```

```

      TT=T+T
      J=J+1
      F1=1/(J)
      F2=0.
      DO 225 I=1,M2
      J=J+1
      TEMPI=F1
      F1=TT+F1-F2+CI1(J)
      F2=TEMPI
225  CONTINUE
      ANS=((T+F1-F2+CI1(J))/SQRT(AX)
      GO TO (226,217),MODE
226  IF(AX.GT.ELIM) GO TO 92
      GO TO 216
90  CONTINUE
      CALL ERRCHK(44,44)IN BES101, SCALING OPTION, MODE, NOT 1 OR 2.)
      RETURN
91  CONTINUE
      CALL ERRCHK(33,33)IN BES101, ORDER, NU, NOT 0 OR 1.)
      RETURN
92  CONTINUE
      CALL ERRCHK(49,49)IN BES101, OVERFLOW, ABS(X) TOO LARGE FOR MODE=1
      1.)
      RETURN
      END
      SUBROUTINE BESJ(X,ALPHA,N,Y,NZ)

```

WRITTEN BY D.E. AMOS, S.L. DANIEL AND M.K. WESTON, JANUARY, 1975.

REFERENCE SAND-75-0147

ABSTRACT
 BESJ COMPUTES AN N MEMBER SEQUENCE OF J BESSEL FUNCTIONS
 $J_{\text{SUB}(\text{ALPHA}+K-1)/(\text{X})}$, $K=1, \dots, N$ FOR NON-NEGATIVE ALPHA AND X.
 A COMBINATION OF THE POWER SERIES, THE ASYMPTOTIC EXPANSION
 FOR X TO INFINITY AND THE UNIFORM ASYMPTOTIC EXPANSION FOR
 NU TO INFINITY ARE APPLIED OVER SUBDIVISIONS OF THE (NU,X)
 PLANE. FOR VALUES OF (NU,X) NOT COVERED BY ONE OF THESE
 FORMULAE, THE ORDER IS INCREMENTED OR DECREMENTED BY INTEGER
 VALUES INTO A REGION WHERE ONE OF THE FORMULAE APPLY. BACKWARD
 RECURSION IS APPLIED TO REDUCE ORDERS BY INTEGER VALUES EXCEPT
 WHERE THE ENTIRE SEQUENCE LIES IN THE OSCILLATORY REGION. IN
 THIS CASE FORWARD RECURSION IS STABLE AND VALUES FROM THE
 ASYMPTOTIC EXPANSION FOR X TO INFINITY START THE RECURSION
 WHEN IT IS EFFICIENT TO DO SO. LEADING TERMS OF THE SERIES AND
 UNIFORM EXPANSION ARE TESTED FOR UNDERFLOW. IF A SEQUENCE IS
 REQUESTED AND THE LAST MEMBER WOULD UNDERFLOW, THE RESULT IS
 SET TO ZERO AND THE NEXT LOWER ORDER TRIED, ETC., UNTIL A
 MEMBER COMES ON SCALE OR ALL MEMBERS ARE SET TO ZERO. OVERFLOW
 CANNOT OCCUR. BESJ CALLS SUBROUTINE JAIRY AND FUNCTION GAMLN.

DESCRIPTION OF ARGUMENTS

```

INPUT
X      - X,GE.0
ALPHA  - ORDER OF FIRST MEMBER OF THE SEQUENCE, ALPHA,GE.0
N      - NUMBER OF MEMBERS IN THE SEQUENCE, N,GE.1

OUTPUT
Y      - A VECTOR WHOSE FIRST N COMPONENTS CONTAIN
        VALUES FOR  $J_{\text{SUB}(\text{ALPHA}+K-1)/(\text{X})}$ ,  $K=1, \dots, N$ 
NZ     - NUMBER OF COMPONENTS OF Y SET TO ZERO DUE TO
        UNDERFLOW.
        NZ=0, NORMAL RETURN, COMPUTATION COMPLETED
        NZ.NE.0, LAST NZ COMPONENTS OF Y SET TO ZERO,
        Y(K)=0.,  $K=N-NZ+1, \dots, N$ .

```


7-9. 18182415432400E+01, 4.25349987453885E+01, -7.36879435947963E+00,
 8.2.108001708984E-01, 5(0.), 2.125701300397E+02,
 9-7. 3252468141182E+02, 1.0599045252800E+03, -6.99579627376E+02,
 A. 2.18190511744212E+02, -2.64914304869515E+01, 5.72501420974731E-01,
 B. 4(0.), 1.9145766231841E+03, 8.06172218173731E+03,
 C-1. 3586550064341E+04, 1.1655333368645E+04, -5.3056497961340E+03,
 D. 1.20090291321635E+03, -1.0809019786395E+02, 1.72772750258446E+00,
 E. 3(0.), 2.0204291330661E+04, -9.69805983866375E+04,
 F. 1.92547001232532E+05, -2.03400177280416E+05, 1.22200464983017E+05,
 G-4. 11926549688976E+04, 7.10951430248936E+03, -4.93915304773088E+02,
 H. 6.07404200127348E+00, 2(0.)

C

DATA C2
 -2. 4291918790551E+05, 1.31176361466298E+06,
 1-2. 99801591853811E+06, 3.76327129765640E+06, -2.81356322658653E+06,
 2. 1.2683652732162E+06, -3.1645172484564E+05, 4.52187689813627E+04,
 3-2. 4598304818121E+03, 2.4380529695561E+01, 1(0.),
 4. 2.8446985307204E+06, -1.9706819118322E+07, 5.0952602426646E+07,
 5-7. 41051482115327E+07, 6.63445122747290E+07, -3.75671766607634E+07,
 6. 1.32887671664218E+07, -2.78561812808645E+06, 3.08186404612662E+05,
 7-1. 38860897537170E+04, 1.10017140269247E+02/

C

DATA ALFA1
 -4. 4444444444444E-03, -9.22077922077922E-04,
 1-8. 84892884932885E-05, 1.65927687832450E-04, 2.46691372741793E-04,
 2. 2.6595559346255E-04, 2.61824297061501E-04, 2.48730437344656E-04,
 3. 2.32721040083232E-04, 2.16362485712365E-04, 2.00738658762752E-04,
 4. 1.86267636637545E-04, 1.73660775917875E-04, 1.81031705929016E-04,
 5. 1.50274774160908E-04, 1.40503497391270E-04, 1.31686816545923E-04,
 6. 1.23667445558253E-04, 1.16405271474738E-04, 1.09798298372713E-04,
 7. 1.03772410422993E-04, 9.8262607836936E-05, 9.32120517249503E-05,
 8. 8.93735541354589E-04, 2.32241745189222E-04, -1.41986273556691E-05,
 A-1. 1644931672043E-04, -1.50803558053049E-04, -1.55121924918096E-04,
 B-1. 4680975664666E-04, 1.33815503667491E-04, -1.19744975684254E-04,
 C-1. 05184319207974E-04, -9.3759549891194E-05, -8.26923045588193E-05,
 D-7. 2937348155221E-05, -6.4404235721015E-05, -5.69611566009369E-05,
 E-5. 47310443030562E-05, -4.4813486800888E-05, 3.8668872717599E-05,
 F-3. 55400532972042E-05, -3.17414256609022E-05, -2.83996793904175E-05,
 G-2. 54522720634871E-05, -2.28459297164725E-05, -2.05352753106481E-05,
 H-1. 84816217627666E-05, -1.66519330021394E-05/

C

DATA ALFA2
 -3. 54211971457744E-04, -1.56161263945159E-04,
 1. 3.04465503594936E-05, 1.30198655773243E-04, 1.67471106699712E-04,
 2. 1.7022587683533E-04, 1.56501427608595E-04, 1.36339170977445E-04,
 3. 1.14866692029835E-04, 9.45869093034688E-05, 7.64498419250898E-05,
 4. 6.07570334965197E-05, 4.74394299290505E-05, 3.62757512005344E-05,
 5. 2.69939714979232E-05, 1.9321038247939E-05, 1.30056674793963E-05,
 6. 7.8262086674497E-06, 3.59257485819352E-06, 1.44040049814252E-07,
 7-2. 65396769679939E-06, -4.91346867098486E-06, -6.72739296091248E-06,
 8-8. 17269379678658E-06, -9.31304715093561E-06, -1.02011418798016E-05,
 9. 3.78194199201773E-04, 2.02471952761816E-04, -6.37938506318862E-05,
 A-2. 38598230603006E-04, -3.10916256027362E-04, -3.13680115247576E-04,
 B-2. 78950273791323E-04, -2.28564082619141E-04, -1.75245280340847E-04,
 C-1. 25544063060690E-04, -8.22982872820208E-05, -4.62860730588116E-05,
 D-1. 72334302366962E-05, 5.60890482304603E-06, 2.31395443148287E-05,
 E. 3.62642743856794E-05, 4.58006124490189E-05, 5.2435529459114E-05,
 F. 5.68396208545815E-05, 5.9434982033104E-05, 6.064786527578422E-05,
 G. 6.08023907788436E-05, 6.01577894539460E-05, 5.89199657344698E-05,
 H. 5.72515823777593E-05, 5.52804375589653E-05/

C

DATA BETA1
 1. 79988721413553E-02, 5.59964911064388E-03,
 1. 2.88501402231133E-03, 1.8009606761054E-03, 1.24753110589199E-03,
 2. 9.22878876572938E-04, 7.14430421727287E-04, 5.71787281789705E-04,
 3. 4.694310077606482E-04, 3.932328354662917E-04, 3.34818889318298E-04,
 4. 2.8952148495752E-04, 2.52211815549573E-04, 2.2260580798883E-04,
 5. 1.97541838035053E-04, 1.76836855019718E-04, 1.59316595661821E-04,
 6. 1.44347930197334E-04, 1.31448068119985E-04, 1.202454444949303E-04,

```

7 1.449144504599E-04, 1.01828770740567E-04, 9.41998224204 E-05,
8 1.130545753834E-05, 8.13466262162801E-05, 7.59002269646 E-05,
9 -1.49282953213429E-03, -8.78204709546389E-04, -5.02916549572035E-04,
A -2.94822138512746E-04, -1.75463996970783E-04, -1.0400855040816E-04,
B -5.96141953046458E-05, -3.12038929076098E-05, -1.26089735980230E-05,
C -2.42892608575730E-07, 8.05996165414274E-06, 1.36507009262147E-05,
D 1.73964125472926E-05, 1.98672978842134E-05, 2.14463263790823E-05,
E 2.2395465932457E-05, 2.2896783814713E-05, 2.3078538981178E-05,
F 2.30321976860909E-05, 2.28236073720349E-05, 2.25005881105292E-05,
G 2.20981015361991E-05, 2.16418427448104E-05, 2.11507649256221E-05,
H 2.06388749782171E-05, 2.01165241997082E-05,
DATA BETA2 / 5.2213076721293E-04,
1 2.79520853992021E-04, 1.52468156198447E-04, 6.93271105657044E-05,
2 1.76258683069991E-05, -1.3574496343289E-05, -3.17972413350427E-05,
3 -4.18861861696693E-05, -4.6904889379141E-05, -4.8766543713787E-05,
4 -4.87010031186735E-05, -4.74755620890087E-05, -4.55813058138628E-05,
5 -4.3330964511268E-05, -4.09230193157750E-05, -3.84822638603221E-05,
6 -3.6085716735411E-05, -3.3779306123367E-05, -3.15888560722110E-05,
7 -2.95269561730807E-05, -2.75978914828336E-05, -2.58006174666864E-05,
8 -2.41308356761280E-05, -2.25823509518346E-05, -2.11479856768913E-05,
9 -4.74817796559960E-04, -4.77864567147321E-04, -3.20390228067038E-04,
A -1.61105016119962E-04, -4.25778101285435E-05, 3.44571294294968E-05,
B 7.97092684075675E-05, 1.03138236708272E-04, 1.12466775262204E-04,
C 1.13103642108481E-04, 1.08651634848774E-04, 1.01437951597662E-04,
D 9.29298396593384E-05, 8.40293133016090E-05, 7.52727991349134E-05,
E 6.68632521975731E-05, 5.92564547323195E-05, 5.22169308826976E-05,
F 4.58539485165361E-05, 4.0148513891487E-05, 3.50481730031328E-05,
G 3.05157993034347E-05, 2.64956119250516E-05, 2.293636333690998E-05,
H 1.97893056664022E-05, 1.70091984636413E-05,

```

C

```

DATA BETA3 / 7.36465810572578E-04, 8.72790805146194E-04,
1 6.22614862373135E-04, 2.85998154194304E-04, 3.84737672879366E-06,
2 -1.879606003636972E-04, -2.97603646594555E-04, -3.45998126832256E-04,
3 -3.53382470916038E-04, -3.3571563575049E-04, -3.04321124789040E-04,
4 -2.66722723047613E-04, -2.27654214122820E-04, -1.89922611854562E-04,
5 -1.55058918599094E-04, -1.2378240761874E-04, -9.62926147717644E-05,
6 -7.2517832771425E-05, -5.22070028895634E-05, -3.50347750511901E-05,
7 -2.0648976103552E-05, -8.70106096849767E-06, 1.136986866675100E-06,
8 9.16426474122779E-08, 1.5647785428873E-05, 2.08223629482467E-05,

```

C

```

DATA GAMA / 6.29980524947437E-01, 2.51984209378975E-01,
1 1.54790300415656E-01, 1.10713062416159E-01, 8.57309395527395E-02,
2 6.97161316958684E-02, 5.86085671893714E-02, 5.04698873536311E-02,
3 4.42600580689155E-02, 3.93720661543510E-02, 3.54283195924455E-02,
4 3.21818857502098E-02, 2.84645240791158E-02, 2.71581677112934E-02,
5 2.5176827273862E-02, 2.34570755306079E-02, 2.19508390134907E-02,
6 2.06210828235846E-02, 1.94388240897861E-02, 1.83810633000863E-02,
7 1.74293213231963E-02, 1.65685837786612E-02, 1.57865285987918E-02,
8 1.50729501494096E-02, 1.44193250839955E-02, 1.3818480575342E-02,

```

C

TEST INPUT ARGUMENTS

C

```

NZ=0
KT=1
IF(N-1) 92,108,109
108 KT=2
109 NN=N
IFIX) 93,110,120
110 IF(ALPHA) 91,114,116
114 Y(1)=1.
IF(N.EQ.1) RETURN
11=2
GO TO 116
116 11=1
118 DO 119 I=1,N
119 Y(1)=0.

```

```

      RET=RN
120 C      NUE
      IF(ALPHA.LT.0.) GO TO 91
C
      DFN=DBLE(FLOAT(N))+DBLE(ALPHA)-1.0+0
      FNU=DFN
      X02=X+.5
      SX02=X02*X02
C
      DECISION TREE FOR REGION WHERE SERIES, ASYMPTOTIC EXPANSION FOR X
      TO INFINITY AND ASYMPTOTIC EXPANSION FOR NU TO INFINITY ARE
      APPLIED.
C
      IF(SX02.LE.(FNU+.1.)) GO TO 850
      TA=AMAX1(20.,FNU)
      IF(X.GT.TA) GO TO 880
      IF(X.GT.12.) GO TO 860
      X02L=ALOG(X02)
      NS=SX02-FNU
      GO TO 852
850 FNU=FNU
      FNP1=FNU+.1.
      X02L=ALOG(X02)
      IS=KT
      IF(X.LE.0.5) GO TO 134
      NS=0
852 DFN=DFN+DBLE(FLOAT(NS))
      FNU=DFN
      FNP1=FNU+.1.
      IS=KT
      IF(N-1+NS.GT.0) IS=3
      GO TO 134
860 NS=AMAX1(36.-FNU,0.)
      DFN=DFN+DBLE(FLOAT(NS))
      FNU=DFN
      IS=KT
      IF(N-1+NS.GT.0) IS=3
      GO TO 130
880 CONTINUE
      RTA=RTA+RTX
      TAU=RTA+FNULIM(KT)
      TA=TAU+FNULIM(KT)
      IF(FNU.LE.TA) GO TO 500
129 FNU=FNU
      IS=KT
C
      UNIFORM ASYMPTOTIC EXPANSION FOR NU TO INFINITY
C
130 CONTINUE
      XX=X/FNU
      W2=1.-XX*XX
      ABW2=ABS(W2)
      RA=RTA+FNULIM(KT)
      IF(ABW2.GT.0.2775) GO TO 200
C
      CASES NEAR X=FN, ABS(1.-(X/FN)**2).LE.0.2775
      COEFFICIENTS OF ASYMPTOTIC EXPANSION BY SERIES
C
      ZETA AND TRUNCATION FOR A(ZETA) AND B(ZETA) SERIES
C
      KMAX IS TRUNCATION INDEX FOR A(ZETA) AND B(ZETA) SERIES=MAX(2,SA)
C
      SA=0.
      IF(ABW2.EQ.0.) GO TO 21
      SA=TOLS/ALOG(ABW2)

```

```

21 SB=SA
DO 22 I=1,S
  KM I)=AMAX1(SA,2.)
  SA=SA+SB
22 CONTINUE
  KB=KMAX(S)
  KLAST=KB-1
  SA=GAMA(KB)
  DO 24 K=1, KLAST
    KB=KB-1
    SA=SA+W2+GAMA(KB)
24 CONTINUE
  Z=W2+SA
  AZ=ABS(Z)
  RTZ=SORT(AZ)
  FN13=FN+CON2
  RTARY=RTZ+FN13
  ARY=RTARY+RTARY
  AZ2=AZ+RTZ+CON1
  ACZ=FN+AZ2
  IF(Z.LE.0.) GO TO 27
  TEST FOR UNDERFLOW, 1.E-280=EXP(-644.), ONE WORD LENGTH
  UP FROM UNDERFLOW LIMIT OF CDC 6600
  IF(ACZ.GT.ELIM2) GO TO 180
  ARY=ARY
27 PHI=SQRT(SORT(SA+SA+SA+SA))
  C
  C
  C
  B(ZETA) FOR S=0
  C
  C
  KB=KMAX(S)
  KLAST=KB-1
  SB=BETA(KB,1)
  DO 23 K=1, KLAST
    KB=KB-1
    SB=SB+W2+BETA(KB,1)
23 CONTINUE
  KSP1=1
  FN2=FN+FN
  RFN2=1./FN2
  RDN=1.
  ASUM=1.
  RELB=TOL*ABS(SB)
  BSUM=SB
  DO 25 KS=1,4
    KSP1=KSP1+1
    RDN=RDN+RFN2
  C
  C
  A(ZETA) AND B(ZETA) FOR S=1,2,3,4
  C
  C
  KB=KMAX(S-KS)
  KLAST=KB-1
  SA=ALFA(KB,KS)
  SB=BETA(KB,KSP1)
  DO 26 K=1, KLAST
    KB=KB-1
    SA=SA+W2+ALFA(KB,KS)
    SB=SB+W2+BETA(KB,KSP1)
26 CONTINUE
  TA=SA+RDN
  TB=SB+RDN
  ASUM=ASUM+TA
  BSUM=BSUM+TB
  IF(ABS(TA).LE.TOL.AND.ABS(TB).LE.RELB) GO TO 152
25 CONTINUE

```

```

152 CONTINUE
BS=-BSUM/(FN*FN13)
GD 400
C
C 200 CONTINUE
TAU=1./RA
T2=1./W2
IF(W2.GE.0.) GO TO 30
C
C CASES FOR (X/FN).GT.SORT(1.2775)
C
AZ32=ABS(RA-ATAN(RA))
ACZ=AZ32*FN
CZ=-ACZ
Z32=1.5-AZ32
RTZ=Z32*CON2
FN13=FN*CON2
RTARY=RTZ*FN13
ARY=-RTARY*RTARY
GO TO 150
30 CONTINUE
C
C CASES FOR (X/FN).LT.SORT(0.7225)
C
AZ32=ABS(ALOG((1.+RA)/XX) -RA)
C
C TEST FOR UNDERFLOW, 1.E-280 = EXP(-644.), ONE WORD LENGTH
C
C UP FROM UNDERFLOW LIMIT OF CDC 6600
C
ACZ=AZ32*FN
CZ=ACZ
IF(ACZ.GT.ELIM2) GO TO 180
Z32=1.5-AZ32
RTZ=Z32*CON2
FN13=FN*CON2
RTARY=RTZ*FN13
ARY=-RTARY*RTARY
150 CONTINUE
PHI=SQRT((RTZ*RTZ)+TAU)
TB=1.
ASUM=1.
TFN=TAU/FN
UPOL(2)=(C(1,1)*T2+C(2,1))*TFN
RCZ=CON1/CZ
CRZ2=CON548*RCZ
BSUM=UPOL(2)*CRZ2
REL8=TOL*ABS(BSUM)
AP=TFN
KS=0
KP1=2
RZDEN=RCZ
DO 155 LR=2,8,2
C
C COMPUTE TWO U POLYNOMIALS FOR NEXT A(ZETA) AND B(ZETA)
C
LRP1=LR+1
DO 101 K=LR,LRP1
KS=KS+1
KP1=KP1+1
S1=C(1,K)
DO 102 J=2,KP1
S1=S1*T2+C(J,K)
102 CONTINUE
AP=AP*TFN
UPOL(KP1)=AP*S1
CR(KS)=BR(KS)*RZDEN

```

```

RZDFN=RZDEN*RCZ
DR( )=AR(KS)*RZDEN
101 CONINUE
SUMA=UPOL(LRP1)
SUMB=UPOL(LR+2)+UPOL(LRP1)*CRZ32
JU=LRP1
DO 151 JR=1,LR
JU=JU-1
SUMA=SUMA+CR(JR)*UPOL(JU)
SUMB=SUMB+DR(JR)*UPOL(JU)
151 CONTINUE
TB=TB
IF(W2*GT.O.) TB=ABS(TB)
ASUM=ASUM+SUMA*TB
BSUM=BSUM+SUMB*TB
IF(ABS(SUMA).LE.TOL.AND.ABS(SUMB).LE.RELB) GO TO 165
165 CONTINUE
TB=RTARY
IF(W2*GT.O.) TB=-TB
BSUM=BSUM/TB
400 CONTINUE
CALL JAIRV(ARY,RTARY,ACZ,AT,DA1)
TEMP(15)=PMI*(A1*ASUM+DA1*BSUM)/FN13
GO TO (401,202,650), IS
402 TEMP(1)=TEMP(3)
KT=1
401 IS=2
DFN=DFN-1,D=0
FN=DFN
GO TO 130
SERIES FOR (X/2)**2.LE.NU+1
134 CONTINUE
GLN=GAMLN(FNP1)
ARG=FN*XO2L-GLN
IF(ARG.LT.-ELIM1) GO TO 123
EARG=EXP(ARG)
300 CONTINUE
S=1,
AK=3,
T2=1,
T=1,
S1=FN
DO 125 K=1,17
S2=T2*S1
T=T-SKO2/S2
S=S+T
IF(ABS(T).LT.TOL) GO TO 127
T2=T2*AK
AK=AK+2,
S1=S1*FN
125 CONTINUE
127 CONTINUE
TEMP(15)=S*EARG
GO TO (301,202,600), IS
301 EARG=EARG*FN/XO2
DFN=DFN-1,D=0
FN=DFN
IS=2
GO TO 300
SET UNDERFLOW VALUE AND UPDATE PARAMETERS
180 Y(NN)=0.

```

```

NW I-1
DA JFN-1.D+0
FN=DFN
IF (NN-1) 170,171,130
171 KT=2
IS=2
GO TO 130
123 Y(NN)=0.
NN=NN-1
FNP1=FN
DFN=DFN-1.D+0
FN=DFN
IF (NN-1) 170,172,173
172 KT=2
IS=2
173 IF(SX02.LE.FNP1) GO TO 133
GO TO 130
133 ARG=ARG-X02L+ALOG(FNP1)
IF(ARG.LT.-ELIM1) GO TO 123
GO TO 134
170 NZ=N-NN
RETURN
C
C BACKWARD RECURSION SECTION
C
202 CONTINUE
NZ=N-NN
IF(KT.EQ.2) GO TO 250
203 CONTINUE
BACKWARD RECUR FROM INDEX ALPHA+NN-1 TO ALPHA
Y(NN)=TEMP(1)
Y(NN-1)=TEMP(2)
IF (NN.EQ.2) RETURN
DX=X
TRX=2.D+0/DX
DTM=DFN+TRX
TM=DTM
K=NN+1
DO 230 I=3,NN
K=K-1
Y(K-2)=TM+Y(K-1)-Y(K)
DTM=DTM-TRX
TM=DTM
230 CONTINUE
RETURN
250 Y(1)=TEMP(2)
RETURN
C
C ASYMPTOTIC EXPANSION FOR X TO INFINITY WITH FORWARD RECURSION IN
C OSCILLATORY REGION X.GT.MAX(20, NU), PROVIDED THE LAST MEMBER
C OF THE SEQUENCE IS ALSO IN THE REGION.
C
500 CONTINUE
IN=ALPHA-TAU+2.
IF(JN.LE.0) GO TO 502
INP1=IN+1
DALPHA=ALPHA-FLOAT(INP1)
KT=1
GO TO 511
502 DALPHA=ALPHA
IN=0
511 IS=KT
512 ARG=X-PIDT-DALPHA-PDF
SA=SIN(ARG)
SB=COS(ARG)
RA=RTTP/RTX

```



```

503      ETX=B.*X
        D( ALPHA
        DX=X+DX
        DTM=DX+DX
        T2=DTM-1.D+0
        T2=T2/ETX
        S2=T2
        RELB=TOL*ABS(T2)
        T1=ETX
        S1=1.
        FN=1.
        AK=B.
        DO 504 K=1,13
        T1=T1+ETX
        FN=FN+AK
        DX=FN
        TRX=DTM-DX
        AP=TRX
        T2=-T2*AP/T1
        S1=S1+T2
        T1=T1+ETX
        AK=AK+B.
        FN=FN+AK
        DX=FN
        TRX=DTM-DX
        AP=TRX
        T2= T2*AP/T1
        S2=S2+T2
        IF(ABS(T2).LE.RELB) GO TO 505
        AK=AK+B.
504      CONTINUE
505      TEMP(15)=RA*(S1+SB-S2*SA)
        GO TO (506,507),15
506      DALPHA=DALPHA+1.
        IS=2
        TB=SA
        SA=SB
        SB=TB
        GO TO 503
C
C      FORWARD RECURSION SECTION
C
507      IF(KT.EQ.2) GO TO 250
        S1=TEMP(1)
        S2=TEMP(2)
        TX=2./X
        TM=DALPHA+TX
        IF(IN.EQ.0) GO TO 520
C
C      FORWARD RECUR TO INDEX ALPHA
C
        DO 510 I=1,IN
        S=S2
        S2=TM+S2-S1
        TM=TM+TX
        S1=S
510      CONTINUE
        IF(INN.EQ.1) GO TO 535
        S=S2
        S2=TM+S2-S1
        TM=TM+TX
        S1=S
520      CONTINUE
C
C      FORWARD RECUR FROM INDEX ALPHA TO ALPHA+N-1
C

```

```

Y(1)=S1
Y(2)=S2
IF (.EQ.2) RETURN
DO -J0 I=3,NN
Y(I)=TM*Y(I-1)-Y(I-2)
TM=TM*TX
530 CONTINUE
RETURN
535 Y(1)=S2
RETURN

C
C BACKWARD RECURSION WITH NORMALIZATION BY
C ASYMPTOTIC EXPANSION FOR NU TO INFINITY OR POWER SERIES.
C
600 CONTINUE
C COMPUTATION OF LAST ORDER FOR SERIES NORMALIZATION
KM=AMAX1(3,-FN,0.)
TFN=FN+FLOAT(KM)
TA=(GLN+TFN-0.9189385332-0.0833333333/TFN)/(TFN+0.5)
TA=XO2L-TA
TB=-(1.-1.5/TFN)/TFN
IN=CE/(-TA+SQR1(TA+TA-CE*TB))+1.5
IN=IN+KM
GO TO 603

650 CONTINUE
C COMPUTATION OF LAST ORDER FOR ASYMPTOTIC EXPANSION NORMALIZATION
GLN=AZ32+RA
IF(ARY.GT.30.) GO TO 675
RDEN=(PP(4)*ARY+PP(3))*ARY+1.
RZDEN=PP(1)*PP(2)*ARY
TA=RZDEN/RDEN
IF(W2.LT.0.10) GO TO 651
TB=GLN/RTARY
GO TO 677

651 TB=(1.259921049+0.1679894730*W2)/FN13
GO TO 677

675 CONTINUE
TA=CON1+TCE/ACZ
TA=((0.0493827160*TA-0.1111111111)*TA+0.6666666667)*TA*ARY
IF(W2.LT.0.10) GO TO 651
TB=GLN/RTARY
IN=TA/TB+1.5
IF(IN.GT.1N1M) GO TO 602
DX=FLOAT(IN)
DTM=DFN+DX
DX=X
TRA=2.D+0/DX
DTM=DTM-TRA
TM=DTM
TA=0.
TB=TOL
KK=1
605 CONTINUE
C BACKWARD RECUR UNINDEXED
C
DO 601 I=1,IN
S=TB
TB=TM*TB-TA
TA=S
DTM=DTM-TRA
TM=DTM
601 CONTINUE
C NORMALIZATION
IF(KK.NE.1) GO TO 604
TA=(TA/TB)*TEMP(3)

```



```

D{*****NSION DA:P(19),DAJN(19),DA(15),DB(18)
D{*****NSION DAK1(14),DAK2(24),DAK3(14)

DATA N1,N2,N3,N4/14,23,19,15/
DATA M1,M2,M3,M4/12,21,17,13/
DATA FPI12,CON1,CON2,CON3,CON4,COM5/
1 1.30899598995795E+00, 8.666666666666667E-01, 5.03154716196777E+00,
2 3.80004598967293E-01, 8.333333333333333E-01, 8.66025403784439E-01/
DATA AK1 / 2.20423090987793E-01, -1.25290242787700E-01,
1 1.03881163359194E-02, 8.22844152006343E-04, -2.34614345891226E-04,
2 1.63824280172116E-05, 3.06902589573189E-07, -1.2962199359332E-07,
3 8.22908158823688E-09, 1.53953368823298E-11, -3.39165465615682E-11,
4 2.03253257423626E-12, -1.10679546097884E-14, -5.16169497785080E-15/
DATA AK2 / 2.74366150869598E-01, 1.12124917399925E-03,
1 1.57339220621190E-03, 4.27427528348750E-04, -1.12124917399925E-03,
2 2.88763171318904E-05, -7.36804232370554E-06, 1.87290209741024E-06,
3 4.75892793962291E-07, 1.21130416955909E-07, -3.09245374270614E-08,
4 7.92454705282654E-09, -2.03902447167914E-09, 5.26863056595742E-10,
5 1.36704767633569E-10, 3.56141039013708E-11, -9.31388296548430E-12,
6 2.44464450473635E-12, -6.43840261990955E-13, 1.70106030559349E-13,
7 4.50760104503281E-14, 1.19774799164811E-14, -3.19077040865066E-15/
DATA AK3 / 2.80271447340791E-01, -1.78127042844379E-03,
1 4.03422579628999E-05, -1.63249965269003E-06, 9.21181482476768E-08,
2 6.52294330229155E-09, 5.47138404576546E-10, -5.24408251800260E-11,
3 5.60477904117209E-12, -6.5637524639313E-13, 8.31285761966247E-14,
4 1.12705134691063E-14, 1.62267976598129E-15, -2.46480324312426E-16/
DATA AJP / 7.78952966437581E-02, -1.84356363456801E-01,
1 3.01412605216174E-02, 3.053422324377608E-02, -4.95424702513079E-03,
2 1.72749552563952E-03, 2.43137637839190E-04, 5.04564777517082E-05,
3 6.16316582695208E-06, -9.03986745510768E-07, 9.70243778355884E-08,
4 1.09639453305205E-08, -1.04716330588766E-09, -9.60359441344646E-11,
5 8.25358789454134E-12, 6.36123439018768E-13, -4.96629614116015E-14,
6 3.29810288929615E-15, 2.35798252031104E-16/
DATA AJN / 3.80497887617242E-02, -2.45319541845546E-01,
1 1.65820623702696E-01, 7.49330045818789E-02, -2.63476288106641E-02,
2 5.92535597304981E-03, 1.44744409589804E-03, 2.18311831322215E-04,
3 4.10662077680304E-05, -4.66874894171766E-06, 7.15218807277160E-07,
4 6.52964770854633E-08, -8.44284027565946E-09, -6.44186158976978E-10,
5 7.20802286505285E-11, 4.72465431717846E-12, -4.66022632547045E-13,
6 2.67762710389189E-14, 2.36161316570019E-15/
DATA A / 4.90275424742791E-01, 1.57647277946204E-03,
1 9.66195963140306E-05, 1.35916080268815E-07, 2.98157342654859E-07,
2 1.86824767559979E-08, -1.03685737667141E-09, 3.28660818434328E-10,
3 2.57091410632780E-11, -2.32357655300677E-12, 9.57523279048255E-13,
4 1.20340828045719E-13, -2.90907716770715E-15, 4.55656454580149E-15,
5 9.99003874810259E-16/
DATA B / 2.78593552803079E-01, -3.52915691882584E-03,
1 2.31149677384994E-05, 4.71371842263560E-06, -1.12415907931333E-07,
2 2.00100301184339E-08, 2.60948075302193E-09, -3.55098136101216E-11,
3 3.50849978423875E-11, 5.83007187954202E-12, -2.04644828753326E-13,
4 1.10529179476742E-13, 2.87724778038775E-14, -2.88205111009939E-15,
5 3.32656311696166E-16/
DATA N1D,N2D,N3D,N4D/14,24,19,15/
DATA M1D,M2D,N3D,N4D/12,22,17,13/

DATA DAK1 / 2.04567842307887E-01, -6.61322739905664E-02,
1 8.49845800989287E-03, 3.12183491556289E-03, -2.70016489829432E-04,
2 6.35636298679387E-06, 3.02397712409509E-06, -2.18311195330089E-07,
3 5.36194289332826E-10, 1.13098035622310E-09, -7.43023834629073E-11,
4 4.28804170826891E-13, 2.23810925754539E-13, -1.39140135641182E-14/

```

DATA DAK2 / 2.93332343883230E-01, -8.06196784743112E-03,
1 2.42540172333140E-03, -6.82297548850235E-04, 1.8578642775191E-04,
2 7457447684059E-05, 1.32090681239497E-05, -3.4952824044 E-06,
3 9.4362451078835E-07, -2.44732671521867E-07, 6.49307837648 E-08,
4 1.72717621501538E-08, 4.60725763604656E-09, -1.23249055291550E-09,
5 3.30620409488102E-10, -8.89252099772401E-11, 2.39773319878298E-11,
6 -6.48013921153450E-12, 1.75510132023731E-12, -4.76303829833637E-13,
7 1.29498241100810E-13, -3.52679622210430E-14, 9.62005151585923E-15,
8 -2.62786314342292E-15/

DATA DAK3 / 2.84675828811349E-01, 2.53073072619080E-03,
1 -4.83481130337976E-05, 1.84907283946343E-06, -1.0141849117857E-07,
2 7.0592563457153E-09, -5.85325291400382E-10, 5.56357688831339E-11,
3 -5.90889034779500E-12, 6.88574353784436E-13, -8.68588256452194E-14,
4 1.17374762617213E-14, -1.68523146510923E-15, 2.55374773097056E-16/

DATA DAKP / 6.53219131311457E-02, -1.20262933688823E-01,
1 9.780102326263823E-03, 1.67948429230505E-02, -1.97146140182132E-03,
2 -8.45560295098867E-04, 9.42889620701976E-05, 2.2587860945475E-05,
3 -2.29067870915987E-06, -3.76343991136919E-07, 3.45663933559565E-08,
4 4.29611332003007E-09, -3.58673691214989E-10, 3.57245881361895E-11,
5 2.72696091066336E-12, 2.26120653095711E-13, -1.58763205238303E-14,
6 -1.12604374485125E-15, 7.31327529515367E-17/

DATA DAKN / 1.08594539632967E-02, 8.53313194857091E-02,
1 -3.15277068113058E-01, -8.78420725294257E-02, 5.53251906976048E-02,
2 9.41674060503241E-03, -3.32187026018996E-03, -4.11157343156826E-04,
3 1.01297326891346E-04, 9.87833682208396E-06, -1.87312969812393E-06,
4 -1.50798500131468E-07, 2.32687669525394E-08, 1.59599917419225E-09,
5 -2.07665922668385E-10, -1.24103350500302E-11, 1.39631765331043E-12,
6 7.39400971155740E-14, -7.32887475627500E-15/

DATA DA / 4.91627321104601E-01, 3.11164930427489E-03,
1 6.23140762854081E-05, -4.61789776172142E-06, -6.13158880534626E-08,
2 2.87295804656520E-08, -1.81959715372117E-09, -1.44752826642035E-10,
3 4.53724043420422E-11, -3.99655065847223E-12, -3.24089119830323E-13,
4 1.62098952568741E-13, -2.40765247974057E-14, 1.69384811284491E-16,
5 8.17900786477396E-16/

DATA DB / -2.77571356944231E-01, 4.44212833419920E-03,
1 -8.42328522190089E-05, -2.58040318418710E-06, 3.42389720217621E-07,
2 -6.24286894709776E-09, -2.36377836844577E-09, 3.16991042656873E-10,
3 -4.40995691658191E-12, -5.18674221093575E-12, 9.64874015137022E-13,
4 -4.90190576608710E-14, -1.7725430678112E-14, 5.55950610442662E-15,
5 -7.11793337579530E-16/

IF(X.LT.0.) GO TO 300
IF(C.GT.5.) GO TO 200
IF(X.GT.1.2) GO TO 150
T=(X+X-1.2)*CDN4
TT = T + T

J=N1
F1=AK1(J)
F2=0.
DO 105 I=1,M1
J=J-1
TEMP1=F1
F1=TT*F1-F2*AK1(J)
F2=TEMP1
105 CONTINUE
AT=F1-F2*AK1(1)

J=N1D
F1=DAK1(J)
F2=0.
DO 106 I=1,M1D
J=J-1

```

      TT = F1
      F1 = T * F1 - F2 + DAK1(J)
      F2 = TEMP1
106  CONTINUE
      DAI = -(T * F1 - F2 + DAK1(1))
      RETURN
C 150 CONTINUE
      T = (X + X - CON2) * CON3
      TT = T + T
      J = N2
      F1 = AK2(J)
      F2 = 0.
      DO 155 I=1,M2
        J = J-1
        TEMP1 = F1
        F1 = TT * F1 - F2 + AK2(J)
        F2 = TEMP1
155  CONTINUE
      RTRX = SORT(RX)
      EC = EXP(-C)
      AI = EC * (T * F1 - F2 + AK2(1)) / RTRX
      J = N2D
      F1 = DAK2(J)
      F2 = 0.
      DO 156 I=1,M2D
        J = J-1
        TEMP1 = F1
        F1 = TT * F1 - F2 + DAK2(J)
        F2 = TEMP1
156  CONTINUE
      DAI = EC * (T * F1 - F2 + DAK2(1)) * RTRX
      RETURN
C 200 CONTINUE
      T = 10. / C - 1.
      TT = T + T
      J = N1
      F1 = AK3(J)
      F2 = 0.
      DO 205 I=1,M1
        J = J-1
        TEMP1 = F1
        F1 = TT * F1 - F2 + AK3(J)
        F2 = TEMP1
205  CONTINUE
      RTRX = SORT(RX)
      EC = EXP(-C)
      AI = EC * (T * F1 - F2 + AK3(1)) / RTRX
      J = N1D
      F1 = DAK3(J)
      F2 = 0.
      DO 206 I=1,M1D
        J = J-1
        TEMP1 = F1
        F1 = TT * F1 - F2 + DAK3(J)
        F2 = TEMP1
206  CONTINUE
      DAI = -RTRX * EC * (T * F1 - F2 + DAK3(1))
      RETURN
C 300 CONTINUE
      IF (C.GT.5.) GO TO 350
      T = 4 * C - 1.
      TT = T + T
      J = N3

```

```

F1=IP(J)
E1=V(J)
F2=...
E2=0.
DO 305 I=1,M3
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2+AJ(P(J))
E1=TT*E1-E2+AJN(J)
F2=TEMP1
E2=TEMP2
305 CONTINUE
A1=(T*E1-E2+AJN(1))-X*(T*F1-F2+AJ(P(1)))
J=N3D
F1=DAJP(J)
E1=DAJN(J)
F2=0.
E2=0.
DO 306 I=1,M3D
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2+DAJP(J)
E1=TT*E1-E2+DAJN(J)
F2=TEMP1
E2=TEMP2
306 CONTINUE
DA1=X*X*(T*F1-F2+DAJP(1))+ (T*E1-E2+DAJN(1))
RETURN

C
350 CONTINUE
T=10./C-1.
TT=T*T
J=N4
F1=A(J)
E1=B(J)
F2=0.
E2=0.
DO 310 I=1,M4
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2+A(J)
E1=TT*E1-E2+B(J)
F2=TEMP1
E2=TEMP2
310 CONTINUE
TEMP1=T*F1-F2+A(1)
TEMP2=T*E1-E2+B(1)
RTRX=SQRT(RX)
CV=C-FR12
CCV=CCS(CV)
SCV=5*IN(CV)
A1=(TEMP1*CCV-TEMP2*SCV)/RTRX
J=N4D
F1=DA(J)
E1=DB(J)
F2=0.
E2=0.
DO 311 I=1,M4D
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2+DA(J)
E1=TT*E1-E2+DB(J)

```

```

F2=TEMP1
E7 MP2
CO, .NUE
TEMP1=T*F1-F2*DA(1)
TEMP2=T*E1-E2*DB(1)
E1=CCV*CONS+.5*SCV
E2=SCV*CONS-.5*CCV
DA1=(TEMP1*E1-TEMP2*E2)*RTRX
RETURN
END
SUBROUTINE BES1(X,ALPHA,KODE,N,Y,NZ)

```

WRITTEN BY D. E. AMOS AND S. L. DANIEL, JANUARY, 1975.

REFERENCE SAND-75-0152

ABSTRACT

BES1 COMPUTES AN N MEMBER SEQUENCE OF J BESSEL FUNCTIONS $I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$, $K=1, \dots, N$ OR SCALED BESSEL FUNCTIONS $\text{EXP}(-X) \cdot I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$, $K=1, \dots, N$ FOR NON-NEGATIVE ALPHA AND X. A COMBINATION OF THE POWER SERIES, THE ASYMPTOTIC EXPANSION FOR X TO INFINITY, AND THE UNIFORM ASYMPTOTIC EXPANSION FOR NU TO INFINITY ARE APPLIED OVER SUBDIVISIONS OF THE (NU, X) PLANE. FOR VALUES NOT COVERED BY ONE OF THESE FORMULAE, THE ORDER IS INCREMENTED BY AN INTEGER SO THAT ONE OF THESE FORMULAE APPLY. BACKWARD RECURSION IS USED TO REDUCE ORDERS BY INTEGER VALUES. THE ASYMPTOTIC EXPANSION FOR X TO INFINITY IS USED ONLY WHEN THE ENTIRE SEQUENCE (SPECIFICALLY THE LAST MEMBER) LIES WITHIN THE REGION COVERED BY THE EXPANSION. LEADING TERMS OF THESE EXPANSIONS ARE USED TO TEST FOR OVER OR UNDERFLOW WHERE APPROPRIATE. IF A SEQUENCE IS REQUESTED AND THE LAST MEMBER WOULD UNDERFLOW, THE RESULT IS SET TO ZERO AND THE NEXT LOWER ORDER TRIED, ETC., UNTIL A MEMBER COMES ON SCALE OR ALL ARE SET TO ZERO. AN OVERFLOW CANNOT OCCUR WITH SCALING. BES1 CALLS FUNCTION GAMLN.

DESCRIPTION OF ARGUMENTS

INPUT
X
ALPHA
KODE
N
Y
NZ

- X.GE.0
- ORDER OF FIRST MEMBER OF THE SEQUENCE. ALPHA.GE.0
- A PARAMETER TO INDICATE THE SCALING OPTION
KODE=1 RETURNS $Y(K) = I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$, $K=1, \dots, N$
KODE=2 RETURNS $Y(K) = \text{EXP}(-X) \cdot I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$, $K=1, \dots, N$
N - NUMBER OF MEMBERS IN THE SEQUENCE, N.GE.1

OUTPUT
Y
NZ

- A VECTOR WHOSE FIRST N COMPONENTS CONTAIN VALUES FOR $I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$ OR SCALED VALUES FOR $\text{EXP}(-X) \cdot I_{\text{SUB}}(\text{ALPHA}+K-1)/(X)$, $K=1, \dots, N$ DEPENDING ON KODE
- NUMBER OF COMPONENTS OF Y SET TO ZERO DUE TO UNDERFLOW.
NZ=0, NORMAL RETURN. COMPUTATION COMPLETED
NZ.NE.0, LAST NZ COMPONENTS OF Y SET TO ZERO, $Y(K)=0$, $K=N-NZ+1, \dots, N$.

ERROR CONDITIONS
IMPROPER INPUT ARGUMENTS - A FATAL ERROR
OVERFLOW WITH KODE=1 - A FATAL ERROR
UNDERFLOW - A NON-FATAL ERROR(NZ.NE.0)


```

C
C      !      USES SUBROUTINES GAMLN, ERRCHK, ERRGET, ERRPT, ERR...
C      BE...
C      COMPIL DECKS BESI, GAMLN, ERRCHK
C
C      DOUBLE PRECISION DX, TRX, DTM, DFN
C      DIMENSION Y(1), TEMP(3)
C      DIMENSION C(11,10)
C      DIMENSION C1(11,8), C2(11,2)
C      EQUIVALENCE (C(1,1), C1(1,1))
C      EQUIVALENCE (C(1,9), C2(1,1))
C
C      DATA ELIM, TOL / 667. / 1.E-15 /
C
C      DATA RTPI, RTTPI / 1.59154943091895E-01, 3.98942280401433E-01 /
C
C      DATA CE / 3.45387763900000E+01 /
C
C      DATA INLM / 80 /
C
C      DATA C1 9(0.) /-2.0833333333333E-01, 1.2500000000000E-01,
1 3.3420138888888E-01, -4.0104166666666E-01,
2 7.0312500000000E-02, 8.9121093750000E-01, -1.02581259645062E+00,
3 1.8464626736111E+00, -8.9121093750000E-01, 7.3242187500000E-02,
4 7(0.) 4.6898442342525E+00, -1.12070026162230E-01,
5 8.78912353515625E+00, -2.36408691406250E+00, 1.12152059609375E-01,
6 6(0.) -2.8212072582002E+01, 8.46362176746007E+01,
7-9.18182415432400E+01, 4.2534987453885E+01, -7.36879435947963E+00,
8 2.27108001708984E-01, 2.12570130039217E+02,
9-7.65252468141182E+02, 1.0599045252800E+03, -6.99579627376133E+02,
A 2.18190511744212E+02, -2.64914304869516E+01, 5.72501420974731E-01,
B 4(0.) -1.91945766231841E+03, 8.0617218173731E+03,
C-1.35865500064341E+04, 1.16553333368645E+04, -5.30564697861340E+03,
D 1.20090291321635E+03, -1.08090919788395E+02, 1.7277250258446E+04,
E 3(0.) 2.02042813309661E+04, -9.69805983886375E+04,
F 1.92547001233532E+05, -2.03400177280416E+05, 1.2200464983017E+05,
G-4.11926549686976E+04, 7.10951430248936E+03, -4.93915304773086E+02,
H 6.07404200127348E+00, 2(0.) /
C
C      DATA C2 /-2.42919187900551E+05, 1.31176361466298E+06,
1-2.99801591853811E+06, 3.76327129765640E+06, -2.81356322659653E+06,
2 1.26836527332162E+06, -3.31645172484564E+05, 4.52187689813627E+04,
3-2.49983048181121E+03, 2.43805296995561E+01, 1(0.)
4 3.28446985307204E+06, -1.97088191184322E+07, 5.09526024926646E+07,
5-7.41051482115327E+07, 8.63445122747290E+07, -3.75671766607834E+07,
6 1.32887671664218E+07, -2.7856181280845E+06, 3.08186404612662E+05,
7-1.38866397537170E+04, 1.10017140269247E+02 /
C
C      TEST INPUT ARGUMENTS
C
C      NZ=0
C      KT=1
C      IF(N=1) 92,108,109
108 KT=2
109 NN=N
C      IF(KODE.LT.1.OR.KODE.GT.2) GO TO 90
110 IF(X) 93,110,120
111 IF(ALPHA) 91,114,116
114 Y(1)=1.
C      IF(N.EQ.1) RETURN
115 I=2
C      GO TO 116
116 I=1
118 DO 119 I=1,N
119 Y(I)=0.

```

```

      RETURN
120 C  IF (ALPHA.LT.0.) GO TO 91
C
      DFN=DBLE(FLOAT(N))+DBLE(ALPHA)-1.D+0
      FNU=DFN
      IN=0
      X02=X*.5
      SX02=X02*X02
      ETA=KODE-1
      SX=ETX*X
C
      DECISION TREE FOR REGION WHERE SERIES, ASYMPTOTIC EXPANSION FOR X
      TO INFINITY AND ASYMPTOTIC EXPANSION FOR NU TO INFINITY ARE
      APPLIED.
C
      IF(SX02.LE.(FNU+1.)) GO TO 850
      IF(X.LE.12.) GO TO 870
      FN=0.55*FNU*FNU
      FN=AMAX1(17.,FN)
      IF(X.GE.FN) GO TO 500
      NS=AMAX1(36.,-FNU,0.)
      DFN=DFN+DBLE(FLOAT(NS))
      FN=DFN
      IS=KT
      KM=N-1+NS
      IF(KM.GT.0) IS=3
      GO TO 129
850 FN=FNU
      FNP1=FN+1.
      X02L=ALOG(X02)
      IS=KT
      IF(X.LE.0.5) GO TO 134
      NS=0
      DFN=DFN+DBLE(FLOAT(NS))
      FN=DFN
      FNP1=FN+1.
      IS=KT
      IF(N-1+NS.GT.0) IS=3
      GO TO 134
870 X02L=ALOG(X02)
      NS=SK02-FNU
      GO TO 852
129 CONTINUE
C
      OVERFLOW TEST ON UNIFORM ASYMPTOTIC EXPANSION
C
      IF(KODE.EQ.2) GO TO 130
155 IF(ALPHA.LT.1.) GO TO 156
      Z=X/ALPHA
      RA=SQRT(1.+Z*Z)
      GLN=ALOG((1.+RA)/Z)
      T=RA*(1.-ETX)+ETX/(Z+RA)
      ARG=ALPHA*(T-GLN)
      IF(ARG.GT.ELIM) GO TO 94
      IF(KM.EQ.0) GO TO 157
130 CONTINUE
C
      UNDERFLOW TEST ON UNIFORM ASYMPTOTIC EXPANSION
C
      Z=X/FN
      RA=SQRT(1.+Z*Z)
      GLN=ALOG((1.+RA)/Z)
      T=RA*(1.-ETX)+ETX/(Z+RA)
      ARG=FN*(T-GLN)
157 IF(ARG.LT.-ELIM) GO TO 180

```

```

GO TO 148
188 IF GT.ELIM) GO TO 94
GC J 130
C
C UNIFORM ASYMPTOTIC EXPANSION FOR NU TO INFINITY
C
199 IF(KM.NE.0) GO TO 198
Y(1)=TEMP(3)
RETURN
198 TEMP(1)=TEMP(3)
IN=NS
KT=1
200 CONTINUE
IS=2
DFN=DFN-1.D+0
FN=DFN
Z=X/FN
RA=SQRT(1.+Z*Z)
GLN=ALOG((1.+RA)/Z)
T=RA*(1.-ETX)*ETX/(Z+RA)
ARG=FN*(T-GLN)
148 COEF=EXP(ARG)
T1=T/T
T2=T/T
T1/FN
S2=1.
AP=1.
DO 140 K=1,10
KPI=K+1
S1=C(1,K)
DO 135 J=2,KPI
S1=S1*T2+C(J,K)
135 CONTINUE
AP=AP*T
AK=AP*S1
S2=S2+AK
IF(ABS(AK).LT.TOL) GO TO 145
140 CONTINUE
145 TEMP(15)=SORT(T*RTPI)*COEF*S2
GO TO (200,202,650), 15
C
C SERIES FOR (X/2)**2.LE.NU+1
C
134 CONTINUE
GLN=GMLN(FNPI)
ARG=FN*LOG(1-GLN-SX)
IF(ARG.LT.-ELIM) GO TO 123
EARG=EXP(ARG)
300 CONTINUE
S=1.
AK=3.
T2=1.
T=1.
S1=FN
DO 125 K=1,17
S2=T2+S1
T1= SXO2/S2
S=S+T1
IF(ABS(T1).LT.TOL) GO TO 127
T2=T2+AK
AK=AK+2.
S1=S1+FN
125 CONTINUE
127 TEMP(15)=S+EARG

```

```

GO TO (301,202,600),IS
301 EA' SARG=FN/X02
   DF1 .FN-1.D+0
   FN=DFN
   IS=2
   GO TO 300
C
C SET UNDERFLOW VALUE AND UPDATE PARAMETERS
C
180 Y(NN)=0.
   NN=NN-1
   DFN=DFN-1.D+0
   FN=DFN
   IF (NN-1) 170,171,130
171 KT=2
   IS=2
   GO TO 130
123 Y(NN)=0.
   NN=NN-1
   FNP1=FN
   DFN=DFN-1.D+0
   FN=DFN
   IF (NN-1) 170,172,173
172 KT=2
   IS=2
173 IF(SX02.LE.FNP1) GO TO 133
   GO TO 130
133 ARG=ARG-X02L+ALOG(FNP1)
   IF(ARG.LT.-ELIM) GO TO 123
   GO TO 134
170 NZ=N-NN
   RETURN
C
C BACKWARD RECURSION SECTION
C
202 CONTINUE
   NZ=N-NN
204 GO TO (203,250), KT
203 CONTINUE
   S1=TEMP(1)
   S2=TEMP(2)
   DX=X
   TRA=2.D+0/DX
   DTM=DFN+TRX
   TM=DTM
   IF(IN.EQ.0) GO TO 220
   BACKWARD RECUR TO INDEX ALPHA+NN-1
   DO 210 I=1,IN
   S=S2
   S2=TM+S2+S1
   S1=S
   DTM=DTM-TRX
   TM=DTM
210 CONTINUE
   Y(NN)=S1
   IF(NN.EQ.1) RETURN
   Y(NN-1)=S2
   IF(NN.EQ.2) RETURN
   GO TO 221
220 CONTINUE
   BACKWARD RECUR FROM INDEX ALPHA+NN-1 TO ALPHA
   Y(NN)=S1
   Y(NN-1)=S2
221 IF(NN.EQ.2) RET'
   K=NN+1
   DO 230 I=3,NN

```

```

K=K-1
Y(L)=TM*Y(K-1)+Y(K)
DFA JTM-TRX
TM=DTM
230 CONTINUE
250 Y(1)=TEMP(2)
RETURN
C
C ASYMPTOTIC EXPANSION FOR X TO INFINITY
C
500 CONTINUE
EARG=RTPL/SQRT(X)
IF(KODE.EQ.2) GO TO 502
501 IF(X.GT.ELIM) GO TO 94
EARG=EARG*EXP(X)
502 ETX=B.*X
IS=KT
IN=O
FN=FN+U
503 DX=DFN+DFN
DTM=DX*DX
S1=ETX
TRX=S1
DX=-(DTM-1.D+O)/TRX
T=DX
TRX=1.D+O + DX
S=TRX
S2=1.
AK=B.
DO 504 K=1,25
S1=S1+ETX
S2=S2+AK
DX=S2
TRX=DTM-DX
AP=TRX
T=T*AP/S1
S=S+T
IF(ABS(T).LE.TOL) GO TO 505
AK=AK+B.
504 CONTINUE
505 TEMP(1S)=S+EARG
GO TO (508,204),IS
508 IS=2
DFN=DFN-1.D+O
FN=DFN
GO TO 503
C
C BACKWARD RECURSION WITH NORMALIZATION BY
C ASYMPTOTIC EXPANSION FOR NU TO INFINITY OR POWER SERIES.
C
600 CONTINUE
C COMPUTATION OF LAST ORDER FOR SERIES NORMALIZATION
NM=AMAX1(3.-FN,0.)
TFN=FN+FLOAT(NM)
TA=(GLN+TFN-0.9189385332-0.0833333333/TFN)/(TFN+0.5)
TA=XO2L-TA
TB=-(1.-1./TFN)/TFN
IN=CE/(-TA+SQRT(TA*TA-CE*TB))+1.5
IN=IN+NM
GO TO 603
650 CONTINUE
C COMPUTATION OF LAST ORDER FOR ASYMPTOTIC EXPANSION NORMALIZATION
IN=CE/(GLN+SQRT(GLN+GLN+1.*CE))+1.5
IF(IN.GT.INLIM) GO TO 199
603 DX=FLOAT(IN)

```

```

D1 1FN+DX
DX
TRX=2.D+0/DX
DTM=DTM-TRX
TM=DTM
TA=0.
TB=TOL
KK=1
605 CONTINUE
C
C BACKWARD RECUR UNINDEXED
C
DO 601 I=1.IN
S=TB
TB=TM-TB+TA
TA=S
DTM=DTM-TRX
TM=DTM
601 CONTINUE
C NORMALIZATION
IF(KK.NE.1) GO TO 604
TA=(TA/TB)*TEMP(3)
TB=TEMP(3)
KK=2
IN=NS
IF(NS.NE.0) GO TO 605
604 Y(NN)=TB
615 NZ=N-NN
IF(NN.EQ.1) RETURN
TB=TM-TB+TA
K=NN-1
Y(K)=TB
IF(NN.EQ.2) RETURN
DTM=DTM-TRX
TM=DTM
KM=K-1
C BACKWARD RECUR INDEXED
C
C
DO 602 I=1.KM
Y(K-1)=TM+Y(K)+Y(K+1)
DTM=DTM-TRX
TM=DTM
K=K-1
602 CONTINUE
RETURN
C
C
90 CONTINUE
CALL ERRCHK(42,42HIN BESI, SCALING OPTION, MODE, NOT 1 OR 2.)
RETURN
91 CONTINUE
CALL ERRCHK(38,38HIN BESI, ORDER, ALPHA, LESS THAN ZERO.)
RETURN
92 CONTINUE
CALL ERRCHK(25,25HIN BESI, N LESS THAN ONE.)
RETURN
93 CONTINUE
CALL ERRCHK(26,26HIN BESI, X LESS THAN ZERO.)
RETURN
94 CONTINUE
CALL ERRCHK(42,42HIN BESI, OVERFLOW, X TOO LARGE FOR MODE=1.)
RETURN
END
SUBROUTINE BESYN(X,NU,N,Y)

```

W/ TEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974.

REFERENCE SAND-75-0150

ABSTRACT

BESYN IMPLEMENTS FORWARD RECURSION ON THE THREE TERM RECURSION RELATION FOR A SEQUENCE OF INTEGER ORDER BESSEL FUNCTIONS $Y/\text{SUB}(NU+K-1)/(X)$, $K=1, \dots, N$ FOR REAL $X \geq 0$ AND A NON-NEGATIVE INTEGER NU . IF $NU \leq \text{LT.NULIM}$, ORDERS 0 AND 1 ARE OBTAINED FROM FUNCTION BESY01 TO START THE RECURSION. IF $NU \geq \text{NULIM}$, THE UNIFORM ASYMPTOTIC EXPANSION IS USED FOR ORDERS NU AND $NU+1$ TO START RECURSION. $\text{NULIM}=100$ RESTRICTS FORWARD RECURSION TO RETAIN ACCURACY. AN OVERFLOW TEST IS MADE ON THE LEADING TERM OF THE ASYMPTOTIC EXPANSION BEFORE ANY EXTENSIVE COMPUTATION IS DONE. BESYN CALLS FUNCTION BESY01 AND SUBROUTINE ASYBES. BESY01 CALLS FUNCTION BESJ01. ASYBES CALLS SUBROUTINE YBAIRY.

DESCRIPTION OF ARGUMENTS

INPUT
 X - $X \geq 0$
 NU - ORDER OF THE INITIAL Y FUNCTION, $NU=0, 1, 2, \dots$
 N - NUMBER OF MEMBERS IN THE SEQUENCE, $N \geq 1$

OUTPUT
 Y - A VECTOR WHOSE FIRST N COMPONENTS CONTAIN VALUES FOR $Y(K)=Y/\text{SUB}(NU+K-1)/(X)$, $K=1, \dots, N$

ERROR CONDITIONS
 IMPROPER INPUT ARGUMENTS - A FATAL ERROR
 OVERFLOW - A FATAL ERROR

BESYN USES SUBROUTINES BESY01, BESJ01, ASYBES, YBAIRY, ERRCHK, ERRGET, ERRPRT, ERRSET, ERNSTCT
 COMPILER DECKS BESYN, BESY01, BESJ01, ERRCHK

DIMENSION Y(1)

DATA NULIM / 100 /

DATA ELIM/667./

TEST INPUT ARGUMENTS

IF(MU.LT.0) GO TO 91
 IF(X.LE.0.) GO TO 92
 IF(N.LT.1) GO TO 93
 NN=MINO(2,N)

FN=NU

FN=NU+N-1

IF(FN.LT.2.) GO TO 300

OVERFLOW TEST (LEADING EXPONENTIAL OF ASYMPTOTIC EXPANSION)
 FOR THE LAST ORDER, $NU+N-1 \geq 2$

XXN=X/FN
 W2N=1.-XXN*XXN
 IF(W2N.LE.0.) GO TO 20
 RAN=SQRT(W2N)
 AZN=ALOG((1.+RAN)/XXN)-RAN
 CN=FN+AZN

```

      IF J.GT.ELIM) GO TO 94
20  IF J.LT.NULIM) GO TO 200
      CALL ASYBES(FNU,NN,X,Y)
      GO TO (25,27).NN
25  RETURN
27  TRA=2./X
      TM=TRX*(FNU+1.)
      GO TO 250

C
200  CONTINUE
      S1=BESY01(X,0,DJ)
      S2=BESY01(X,1,DJ)
      TRA=2./X
      TM=TRX
      IN=NU-1
      IF(IN) 220,211,203
      FORWARD RECUR FROM 0 TO NU TO GET Y(1)
C 203 DO 210 I=1,IN
      S=S2
      S2=TM+S2-S1
      S1=S
      TM=TM+TRX
210  CONTINUE
      FORWARD RECUR FROM NU TO NU+1 TO GET Y(2)
C 211 IF(N.EQ.1) GO TO 251
      S=S2
      S2=TM+S2-S1
      S1=S
      TM=TM+TRX
220  CONTINUE
      FORWARD RECUR FROM NU+2 TO NU+N-1
      Y(1)=S1
      Y(2)=S2
250  CONTINUE
      IF(N.EQ.2) RETURN
      NM2=N-2
      DO 230 I=1,NM2
      Y(I+2)=TM+Y(I+1)-Y(I)
      TM=TM+TRX
230  CONTINUE
      RETURN
251  Y(1)=S2
      RETURN

C
300  CONTINUE
      NUM1=NU-1
      DO 315 I=1,N
      J=NUM1+I
      ANS=BESY01(X,J,DJ)
      Y(I)=ANS
315  CONTINUE
      RETURN

C
C
C
91  CONTINUE
      CALL ERRCHK(36,36HIN BESYN, ORDER, NU, LESS THAN ZERO.)
      RETURN
92  CONTINUE
      CALL ERRCHK(39,39HIN BESYN, X LESS THAN OR EQUAL TO ZERO.)
      RETURN
93  CONTINUE
      CALL ERRCHK(26,26HIN BESYN, N LESS THAN ONE.)
      RETURN
94  CONTINUE

```


AD-A139 604

MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND
ELECTRON TRANSPORT VOLUME 2(U) ARCON CORP WALTHAM MA
S WOOLF ET AL JAN 84 RADC-TR-83-313-VOL-2

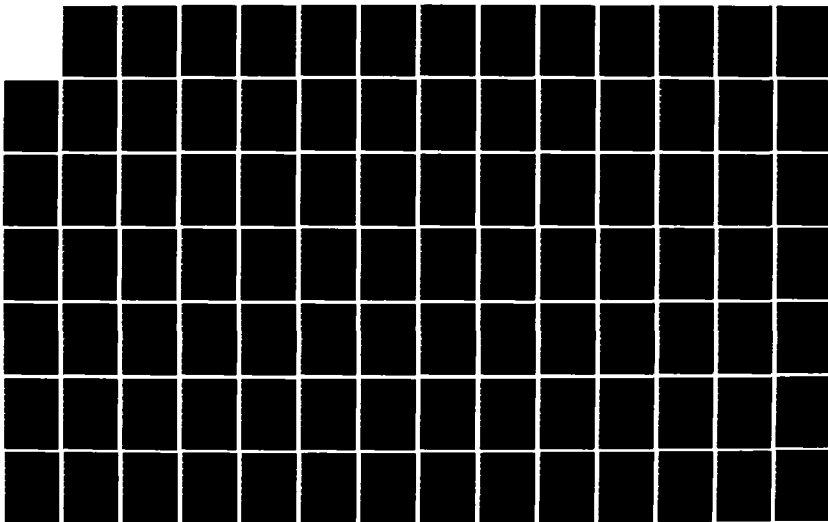
3/5

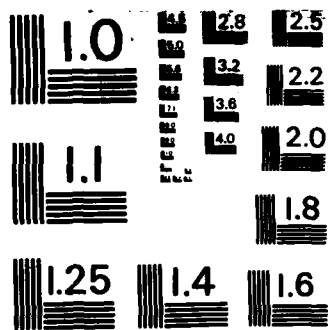
UNCLASSIFIED

F19628-78-C-0188

F/G 9/2

NL





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

```

CALL ERRCHK(53.53HIN BESYN, OVERFLOW, NU OR N TOO LARGE OR X TOO S
1MAY' )
RE: A
END
SUBROUTINE ASYBES(FNU,IN,X,Y)

```

```

      ASYBES COMPUTES Y BESSEL FUNCTIONS
      Y/SUB(NU+K-1)/(X), K=1,IN
      FOR X.GT.0., FNU.GE.32. AND IN=1,2

```

INPUT

```

FNU - ORDER OF FIRST BESSEL FUNCTION
IN  - NUMBER OF Y FUNCTIONS DESIRED, IN=1 OR 2
X   - ARGUMENT, X.GT.0

```

OUTPUT

```

Y - A VECTOR WHOSE FIRST IN COMPONENTS CONTAIN THE SEQUENCE
   Y(K)=Y/SUB(NU+K-1)/(X), K=1,IN

```

WRITTEN BY
D. E. AMOS
S. L. DANIEL

ABSTRACT

ASYBES IMPLEMENTS THE UNIFORM ASYMPTOTIC EXPANSION OF
Y/SUB(FNU)/(4) FOR FNU.GE.32 AND REAL X.GT.0. THE FORM, EXCEPT
FOR SIGN, IS IDENTICAL TO THAT IMPLEMENTED IN JBESS WHERE
THE AI(X) AIRY FUNCTION IS REPLACED BY THE BI(X) FUNCTION.
CORRESPONDINGLY, YBAIRY REPLACES JAIRY FOR THE AIRY FUNCTION
EVALUATIONS. ASYBES CALLS YBAIRY.

```

DIMENSION Y(1)
DIMENSION C(11,10),ALFA(26,4),BETA(26,5)
DIMENSION C1(11,8),C2(11,2),ALFA1(26,2),ALFA2(26,2)
DIMENSION BETA1(26,2),BETA2(26,2),BETA3(26,1)
DIMENSION GAMMA(26),KMAX(5),AR(8),BR(10),UPOL(10)
DIMENSION CR(10),DR(10)

EQUIVALENCE(C(1,1),C1(1,1))
EQUIVALENCE(C(1,9),C2(1,1))
EQUIVALENCE(ALFA(1,1),ALFA1(1,1))
EQUIVALENCE(ALFA(1,3),ALFA2(1,1))
EQUIVALENCE(BETA(1,1),BETA1(1,1))
EQUIVALENCE(BETA(1,3),BETA2(1,1))
EQUIVALENCE(BETA(1,5),BETA3(1,1))

DATA ELIM1,ELIM2,FNULIM,TOL/ 667. , 644. , 32. , 1.E-15 /
DATA TOLS=LN(1.E-3)
DATA TOLS /-6.9077527898214E+00/

DATA UPOL(1),CON1,CON2,CON3,CON548 / 1.000000000000000E+00,
1 6.666666666666667E-01, 3.333333333333333E-01, 1.41421356237310E+00,
2 1.041666666666667E-01/

DATA AR / 8.35503472222222E-02, 1.28226574556327E-01,
1 2.91849026464140E-01, 8.81627267443758E-01, 3.32140828186277E+00,
2 1.49957629868626E+01, 7.89230130115865E+01, 4.74451538868264E+02/

DATA BR /-1.45833333333333E-01,-9.87413194444444E-02,
1-1.43312053915895E-01,-3.17227202678414E-01,-9.42428147957120E-01,
2-3.51120304082638E+00,-1.57272636203680E+01,-8.22814390971859E+01,
3-4.92355370523671E+02,-3.31621856854797E+03/

DATA C1 /-2.08333333333333E-01, 1.25000000000000E-01,

```


1 2.88501402231133E-03, 1.8008606761054E-03, 1.24753110589199E-03,
 2 9.878876572938E-04, 7.1443042172787E-04, 5.717872817891E-04,
 3 4.431007606482E-04, 3.93232835462917E-04, 3.348188893181E-04,
 4 2.88952148495752E-04, 2.52211615549573E-04, 2.2280580798883E-04,
 5 1.97541838033063E-04, 1.7583685019718E-04, 1.59316939661821E-04,
 6 4.4347930197334E-04, 1.31448068119965E-04, 1.20245444949303E-04,
 7 1.10449144504599E-04, 1.01828770740567E-04, 9.4199822404238E-05,
 8 7.413054573834E-05, 8.13466262162801E-05, 7.59002269646219E-05,
 9 1.49282953213429E-03, 8.78204709546389E-04, 5.02916549572035E-04,
 A 2.84822138512746E-04, 1.7583996970783E-04, 1.040855046081E-04,
 B 5.56141953048458E-05, 3.12038929076098E-05, 1.26089739880230E-05,
 C 2.48982608575730E-07, 8.05996165414274E-06, 1.36507009267147E-05,
 D 1.73964125472928E-05, 1.98672978842134E-05, 2.1463263790823E-05,
 E 2.23954659232457E-05, 2.28967783814713E-05, 2.30785389811078E-05,
 F 2.3032197680809E-05, 2.28236073720349E-05, 2.25005881105292E-05,
 G 2.20981015361991E-05, 2.16418427448104E-05, 2.11507649256221E-05,
 H 2.06388749782171E-05, 2.01165241997082E-05/

C

DATA BETA2
 1 2.79526653992021E-04, 1.52468156198447E-04, 4.47932581553385E-04,
 2 1.76258683069991E-05, 1.3574498634269E-05, 6.93271105657044E-05,
 3 4.18861861696693E-05, 4.69004889379141E-05, 4.87665447413787E-05,
 4 4.87010031186735E-05, 4.74755620890087E-05, 4.55813058138628E-05,
 5 4.33309644511266E-05, 4.09230193157750E-05, 3.84822638603221E-05,
 6 3.60857167535411E-05, 3.37793306123367E-05, 3.15888560772110E-05,
 7 2.95269561750807E-05, 2.7597891482836E-05, 2.58008174666884E-05,
 8 2.41308356761280E-05, 2.25823509518346E-05, 2.11479656768913E-05,
 9 4.7461779855960E-04, 4.77864507147321E-04, 3.20390228067038E-04,
 A 1.61105018119962E-04, 4.25778101285435E-05, 3.44571294294868E-05,
 B 7.97092684075675E-05, 1.03138236708272E-04, 1.12466775262204E-04,
 C 1.13103642108481E-04, 1.08651634848774E-04, 1.01437951597662E-04,
 D 9.2929839659364E-05, 8.40293133016090E-05, 7.52727991349134E-05,
 E 6.96632521975731E-05, 5.92564547323195E-05, 5.22169308826978E-05,
 F 4.58539485165361E-05, 4.01445513891487E-05, 3.50481730031328E-05,
 G 3.05157995034347E-05, 2.64956119950516E-05, 2.29363633690998E-05,
 H 1.97893056664022E-05, 1.70091984636413E-05/

C

DATA BETA3
 1 6.22614862573135E-04, 2.85998154194304E-04, 8.72790805146194E-04,
 2 1.87906003636972E-04, 2.9760346584555E-04, 3.84737672879366E-06,
 3 3.53382470916038E-04, 3.35715635775049E-04, 3.45998126832656E-04,
 4 2.66722723047613E-04, 2.27654214122820E-04, 3.0432112789040E-04,
 5 1.55058918599094E-04, 1.23778240761874E-04, 9.62926147717644E-05,
 6 7.25178327714425E-05, 5.2207002895634E-05, 3.50347750511901E-05,
 7 2.06489761035552E-05, 8.70106096849767E-06, 1.13698686675100E-06,
 8 9.16426474122779E-06, 1.56477785428873E-05, 2.08223629482467E-05/

C

DATA GAMA
 1 5.4790300415656E-01, 1.70713062416159E-01, 2.51984209978975E-01,
 2 6.9716131695868E-02, 5.86085671893714E-02, 8.57308395527395E-02,
 3 4.42600580699155E-02, 3.93702661543510E-02, 5.04698873536311E-02,
 4 3.21818857502098E-02, 2.94646240791158E-02, 3.54283195924455E-02,
 5 2.51768272973862E-02, 2.34570755306079E-02, 2.19508390134907E-02,
 6 2.06210828235646E-02, 1.94388240897881E-02, 1.83810633800683E-02,
 7 1.74293213231963E-02, 1.65885837766612E-02, 1.57865285987919E-02,
 8 1.50729501494086E-02, 1.44193250839955E-02, 1.38184805735342E-02/

C

130 CONTINUE

FN=FNU

DO 100 JN=1, IN

XX=X/FN

W2=1.-XX*XX

ABW2=ABS(W2)

IF (ABW2.GT.0.2775) GO TO 200

C


```

KB=KB-1
S7  \W2+ALPA(KB,KS)
S8  J*W2+BETA(KB,NSP1)
26 CONTINUE
TA=SA+RDEN
TB=SB+RDEN
ASUM=ASUM+TA
BSUM=BSUM+TB
IF(ABS(TA).LE.TOL.AND.ABS(TB).LE.RELB) GO TO 152
25 CONTINUE
152 CONTINUE
BSUM=BSUM/(FN*FN13)
GO TO 400

C 200 CONTINUE
RA=SQRT(ABW2)
TAU=1./RA
T2=1./W2
IF(W2.GE.0.) GO TO 30
C CASES FOR (X/FN).GT.SQRT(1.2775)
AZ32=ABS(RA-ATAN(RA))
ACZ=AZ32*FN
CZ=-ACZ
Z32=1.5*AZ32
RTZ=Z32*CON2
FN13=FN*CON2
RTARY=RTZ*FN13
ARY=-RTARY*RTARY
GO TO 150
30 CONTINUE

C CASES FOR (X/FN).LT.SQRT(0.7225)
AZ32=ABS(ALOG((1.+RA)/XX) -RA)
TEST FOR OVERFLOW
ACZ=AZ32*FN
CZ=ACZ
IF(ACZ.GT.ELIM1) GO TO 94
Z32=1.5*AZ32
RTZ=Z32*CON2
FN13=FN*CON2
RTARY=RTZ*FN13
ARY=-RTARY*RTARY
150 CONTINUE
PHI=SQRT((RTZ+RTZ)*TAU)
TB=1.
ASUM=1.
TFN=TAU/FN
UPOL(2)=(C(1,1)*T2+C(2,1))*TFN
RCZ=CON1/CZ
CRZ32=CON548*RCZ
BSUM=UPOL(2)*CRZ32
RELB=TOL*ABS(BSUM)
AP=TFN
NS=0
NP1=2
RZDEN=RCZ
DO 155 LR=2,6,2
C COMPUTE TWO U POLYNOMIALS FOR NEXT A(ZE/A) AND B(ZETA)
C
LRP1=LR+1

```

```

DO 101 K=LR,LRP1
  KS  +1
  KP1=AP1+1
  S1=C(1,K)
  DO 102 J=2,KP1
    S1=S1+2*C(J,K)
  102 CONTINUE
  AP=AP+TFN
  UPOL(KP1)=AP+S1
  CR(KS)=BR(KS)*RZDEN
  RZDEN=RZDEN*RCZ
  DR(KS)=AR(KS)*RZDEN
  101 CONTINUE
  SUMA=UPOL(LRP1)
  SUMB=UPOL(LR+2)*UPOL(LRP1)*CR232
  JU=LRP1
  DO 151 JR=1,LR
    JU=JU-1
    SUMA=SUMA+CR(JR)*UPOL(JU)
    SUMB=SUMB+DR(JR)*UPOL(JU)
  151 CONTINUE
  TB=-TB
  IF(W2.GT.O.) TB=ABS(TB)
  ASUM=ASUM+SUMA*TB
  BSUM=BSUM+SUMB*TB
  IF(ABS(SUMA).LE.TOL.AND.ABS(SUMB).LE.REL8) GO TO 165
  155 CONTINUE
  165 TB=RTARY
  IF(W2.GT.O.) TB=-TB
  BSUM=BSUM/TB
C
  400 CONTINUE
  CALL YBAIRY(ARY,RTARY,ACZ,BI,DBI)
  Y(JN)=PHI*(BI*ASUM+DBI*BSUM)/FN13
  FN=FN+1
  100 CONTINUE
  94 RETURN
  CALL ERRCHK(20,20*IN ASYBES, OVERFLOW.)
  RETURN
  END
  SUBROUTINE YBAIRY(X,RX,C,BI,DBI)
    YBAIRY COMPUTES THE AIRY FUNCTION BI(X)
    AND ITS DERIVATIVE DBI(X) FOR ASYBES
    INPUT
    X - ARGUMENT, COMPUTED BY ASYBES, X UNRESTRICTED
    RX - RX=SQRT(ABS(X)), COMPUTED BY ASYBES
    C - C=2.*(ABS(X)**1.5)/3., COMPUTED BY ASYBES
    OUTPUT
    BI - VALUE OF FUNCTION BI(X)
    DBI - VALUE OF THE DERIVATIVE DBI(X)
    WRITTEN BY
    D. E. AMOS
    S. L. DANIEL
    DIMENSION BK1(20),BK2(20),BK3(20),BK4(14)
    DIMENSION BJP(19),BJN(19),AA(14),BB(14)
    DIMENSION DBK1(21),DBK2(20),DBK3(20),DBK4(14)
    DIMENSION DBJP(19),DBJN(19),DAA(14),DBB(14)
C

```


DATA M1,M2,N3/20,19,14/
DA M1,M2,N3/18,17,12/
DA MID,N2D,N3D,M4D/21,20,19,14/
DATA MID,N2D,N3D,M4D/19,18,17,12/
DATA FPI12,SPI12,CON1,CON2,CON3/
1 1.3099593899575E+00, 1.83259571459405E+00, 6.66666666666667E-01,
2 7.74148278841779E+00, 3.64786105490356E-01/
DATA BK1 / 2.43202846447449E+00, 2.57132009754685E+00,
1 1.02802341258618E+00, 3.4195817825872E-01, 8.4197862888284E-02,
2 1.93077282587962E-02, 3.92687837130335E-03, 6.83302689948045E-04,
3 1.14611403991141E-04, 1.741951788337086E-05, 2.41223629958355E-06,
4 3.2452591983273E-07, 4.03508788540183E-08, 4.70875059642298E-09,
5 5.3536743258589E-10, 5.70606721846334E-11, 5.80526363709933E-12,
6 5.76338988616388E-13, 5.42103834518071E-14, 4.91857330301677E-15/
DATA BK2 / 5.74830555784088E-01, -6.91648648376891E-03,
1 1.97460263052093E-03, -5.24043043868923E-04, 1.23965147239661E-04,
2 -2.27059514462173E-05, 2.2357555008526E-06, 4.15174955023899E-07,
3 -2.84985752198231E-07, 8.501871775435E-08, 1.70400828891326E-08,
4 2.25479746746899E-09, -1.09524166577443E-10, -3.41063845099711E-11,
5 1.11262893886662E-11, -1.75542944241734E-12, 1.36298606401767E-13,
6 8.76342105755664E-15, -4.64063099187041E-15, 7.78772758732960E-16/
DATA BK3 / 5.8677703506912E-01, 2.63672828349579E-03,
1 5.12303351473130E-05, 2.10229231564492E-06, 1.42217095113890E-07,
2 1.2852429591264E-08, 7.28556219407507E-10, -3.45236157301011E-10,
3 -2.11919115912724E-10, -8.5880389292376E-11, -8.14873160315074E-12,
4 3.03177845832183E-12, 1.73447220554115E-12, 1.67935548701554E-13,
5 -1.4982288806719E-13, -5.15470458953407E-14, 8.75741841857830E-15,
6 7.9673555352520E-15, -1.29586137861742E-16, -1.187879447520E-15/
DATA BK4 / 4.8544386705114E-01, -3.08525088408463E-03,
1 6.98748404837928E-05, -2.82757234179768E-06, 1.59553313064138E-07,
2 -1.12980692144601E-08, 9.47671515498754E-10, -9.08301736026423E-11,
3 9.70776206450724E-12, -1.13687527254574E-12, 1.43982917533415E-13,
4 -1.95211019558815E-14, 2.01056379909357E-15, -4.26916444775176E-16/
DATA BNP / 1.34918611457638E-01, -3.19314588205813E-01,
1 5.22061846276114E-02, 5.2888912170312E-02, -8.58100756077350E-03,
2 -2.99211002025555E-03, 4.21126741969759E-04, 8.73931820369273E-05,
3 1.06749163477533E-05, -1.56575097253492E-06, 1.68051151983999E-07,
4 1.06901103638691E-08, -1.81374004961922E-09, -1.66339134593739E-10,
5 1.42956335780810E-11, 1.10179811626595E-12, -8.60187724192263E-14,
6 -5.71248177285064E-15, 4.08414552853803E-16/
DATA BUN / 6.59041673525697E-02, -4.24905910566004E-01,
1 2.87209745195830E-01, 1.2978771099606E-01, -4.56354317590358E-02,
2 -1.02630175982540E-02, 2.50704671521101E-03, 3.78127183743483E-04,
3 -7.1128758384084E-05, -8.08651210688923E-06, 1.2387953127285E-06,
4 1.13096815867279E-07, -1.48234283178310E-08, -1.11576315888077E-09,
5 1.24846618243897E-10, 8.18334132555274E-12, -8.07174877048464E-13,
6 -4.63778618768425E-14, 4.09043399081631E-15/
DATA AA / -2.78593552803079E-01, 3.52915691882584E-03,
1 2.31149677384994E-05, -4.71317842283560E-06, 1.12415907931333E-07,
2 2.00100301184339E-08, -2.60948075302193E-09, 3.55098136101216E-11,
3 3.50849978423875E-11, -5.83007187954202E-12, 2.04644828753326E-13,
4 1.10529179476742E-13, -2.87724778038775E-14, 2.88205111009939E-15/
DATA BB / -4.90275424742791E-01, -1.57647277946204E-03,
1 9.66195963140306E-05, -1.3591680268815E-07, -2.9815734254859E-07,
2 1.88824767559978E-08, 1.03685737657141E-09, -3.28660818434328E-10,
3 2.57091410832780E-11, 2.3235765300677E-12, -8.5752327904855E-13,
4 1.20340828049719E-13, 2.90907716770715E-15, -4.55656454560149E-16/
DATA BK1 / 2.43202846447449E+00, 2.57132009754685E+00,
1 1.02802341258618E+00, 3.4195817825872E-01, 8.4197862888284E-02,
2 1.93077282587962E-02, 3.92687837130335E-03, 6.83302689948045E-04,
3 1.14611403991141E-04, 1.741951788337086E-05, 2.41223629958355E-06,
4 3.2452591983273E-07, 4.03508788540183E-08, 4.70875059642298E-09,
5 5.3536743258589E-10, 5.70606721846334E-11, 5.80526363709933E-12,
6 5.76338988616388E-13, 5.42103834518071E-14, 4.91857330301677E-15/
DATA BK2 / 5.74830555784088E-01, -6.91648648376891E-03,
1 1.97460263052093E-03, -5.24043043868923E-04, 1.23965147239661E-04,
2 -2.27059514462173E-05, 2.2357555008526E-06, 4.15174955023899E-07,
3 -2.84985752198231E-07, 8.501871775435E-08, 1.70400828891326E-08,
4 2.25479746746899E-09, -1.09524166577443E-10, -3.41063845099711E-11,
5 1.11262893886662E-11, -1.75542944241734E-12, 1.36298606401767E-13,
6 8.76342105755664E-15, -4.64063099187041E-15, 7.78772758732960E-16/
DATA BK3 / 5.8677703506912E-01, 2.63672828349579E-03,
1 5.12303351473130E-05, 2.10229231564492E-06, 1.42217095113890E-07,
2 1.2852429591264E-08, 7.28556219407507E-10, -3.45236157301011E-10,
3 -2.11919115912724E-10, -8.5880389292376E-11, -8.14873160315074E-12,
4 3.03177845832183E-12, 1.73447220554115E-12, 1.67935548701554E-13,
5 -1.4982288806719E-13, -5.15470458953407E-14, 8.75741841857830E-15,
6 7.9673555352520E-15, -1.29586137861742E-16, -1.187879447520E-15/
DATA BK4 / 4.8544386705114E-01, -3.08525088408463E-03,
1 6.98748404837928E-05, -2.82757234179768E-06, 1.59553313064138E-07,
2 -1.12980692144601E-08, 9.47671515498754E-10, -9.08301736026423E-11,
3 9.70776206450724E-12, -1.13687527254574E-12, 1.43982917533415E-13,
4 -1.95211019558815E-14, 2.01056379909357E-15, -4.26916444775176E-16/
DATA BNP / 1.34918611457638E-01, -3.19314588205813E-01,
1 5.22061846276114E-02, 5.2888912170312E-02, -8.58100756077350E-03,
2 -2.99211002025555E-03, 4.21126741969759E-04, 8.73931820369273E-05,
3 1.06749163477533E-05, -1.56575097253492E-06, 1.68051151983999E-07,
4 1.06901103638691E-08, -1.81374004961922E-09, -1.66339134593739E-10,
5 1.42956335780810E-11, 1.10179811626595E-12, -8.60187724192263E-14,
6 -5.71248177285064E-15, 4.08414552853803E-16/
DATA BUN / 6.59041673525697E-02, -4.24905910566004E-01,
1 2.87209745195830E-01, 1.2978771099606E-01, -4.56354317590358E-02,
2 -1.02630175982540E-02, 2.50704671521101E-03, 3.78127183743483E-04,
3 -7.1128758384084E-05, -8.08651210688923E-06, 1.2387953127285E-06,
4 1.13096815867279E-07, -1.48234283178310E-08, -1.11576315888077E-09,
5 1.24846618243897E-10, 8.18334132555274E-12, -8.07174877048464E-13,
6 -4.63778618768425E-14, 4.09043399081631E-15/
DATA AA / -2.78593552803079E-01, 3.52915691882584E-03,
1 2.31149677384994E-05, -4.71317842283560E-06, 1.12415907931333E-07,
2 2.00100301184339E-08, -2.60948075302193E-09, 3.55098136101216E-11,
3 3.50849978423875E-11, -5.83007187954202E-12, 2.04644828753326E-13,
4 1.10529179476742E-13, -2.87724778038775E-14, 2.88205111009939E-15/
DATA BB / -4.90275424742791E-01, -1.57647277946204E-03,
1 9.66195963140306E-05, -1.3591680268815E-07, -2.9815734254859E-07,
2 1.88824767559978E-08, 1.03685737657141E-09, -3.28660818434328E-10,
3 2.57091410832780E-11, 2.3235765300677E-12, -8.5752327904855E-13,
4 1.20340828049719E-13, 2.90907716770715E-15, -4.55656454560149E-16/
DATA BK1 / 2.43202846447449E+00, 2.57132009754685E+00,
1 1.02802341258618E+00, 3.4195817825872E-01, 8.4197862888284E-02,
2 1.93077282587962E-02, 3.92687837130335E-03, 6.83302689948045E-04,
3 1.14611403991141E-04, 1.741951788337086E-05, 2.41223629958355E-06,
4 3.2452591983273E-07, 4.03508788540183E-08, 4.70875059642298E-09,
5 5.3536743258589E-10, 5.70606721846334E-11, 5.80526363709933E-12,
6 5.76338988616388E-13, 5.42103834518071E-14, 4.91857330301677E-15/
DATA BK2 / 5.74830555784088E-01, -6.91648648376891E-03,
1 1.97460263052093E-03, -5.24043043868923E-04, 1.23965147239661E-04,
2 -2.27059514462173E-05, 2.2357555008526E-06, 4.15174955023899E-07,
3 -2.84985752198231E-07, 8.501871775435E-08, 1.70400828891326E-08,
4 2.25479746746899E-09, -1.09524166577443E-10, -3.41063845099711E-11,
5 1.11262893886662E-11, -1.75542944241734E-12, 1.36298606401767E-13,
6 8.76342105755664E-15, -4.64063099187041E-15, 7.78772758732960E-16/
DATA BK3 / 5.8677703506912E-01, 2.63672828349579E-03,
1 5.12303351473130E-05, 2.10229231564492E-06, 1.42217095113890E-07,
2 1.2852429591264E-08, 7.28556219407507E-10, -3.45236157301011E-10,
3 -2.11919115912724E-10, -8.5880389292376E-11, -8.14873160315074E-12,
4 3.03177845832183E-12, 1.73447220554115E-12, 1.67935548701554E-13,
5 -1.4982288806719E-13, -5.15470458953407E-14, 8.75741841857830E-15,
6 7.9673555352520E-15, -1.29586137861742E-16, -1.187879447520E-15/
DATA BK4 / 4.8544386705114E-01, -3.08525088408463E-03,
1 6.98748404837928E-05, -2.82757234179768E-06, 1.59553313064138E-07,
2 -1.12980692144601E-08, 9.47671515498754E-10, -9.08301736026423E-11,
3 9.70776206450724E-12, -1.13687527254574E-12, 1.43982917533415E-13,
4 -1.95211019558815E-14, 2.01056379909357E-15, -4.26916444775176E-16/
DATA BNP / 1.34918611457638E-01, -3.19314588205813E-01,
1 5.22061846276114E-02, 5.2888912170312E-02, -8.58100756077350E-03,
2 -2.99211002025555E-03, 4.21126741969759E-04, 8.73931820369273E-05,
3 1.06749163477533E-05, -1.56575097253492E-06, 1.68051151983999E-07,
4 1.06901103638691E-08, -1.81374004961922E-09, -1.66339134593739E-10,
5 1.42956335780810E-11, 1.10179811626595E-12, -8.60187724192263E-14,
6 -5.71248177285064E-15, 4.08414552853803E-16/
DATA BUN / 6.59041673525697E-02, -4.24905910566004E-01,
1 2.87209745195830E-01, 1.2978771099606E-01, -4.56354317590358E-02,
2 -1.02630175982540E-02, 2.50704671521101E-03, 3.78127183743483E-04,
3 -7.1128758384084E-05, -8.08651210688923E-06, 1.2387953127285E-06,
4 1.13096815867279E-07, -1.48234283178310E-08, -1.11576315888077E-09,
5 1.24846618243897E-10, 8.18334132555274E-12, -8.07174877048464E-13,
6 -4.63778618768425E-14, 4.09043399081631E-15/
DATA AA / -2.78593552803079E-01, 3.52915691882584E-03,
1 2.31149677384994E-05, -4.71317842283560E-06, 1.12415907931333E-07,
2 2.00100301184339E-08, -2.60948075302193E-09, 3.55098136101216E-11,
3 3.50849978423875E-11, -5.83007187954202E-12, 2.04644828753326E-13,
4 1.10529179476742E-13, -2.87724778038775E-14, 2.88205111009939E-15/
DATA BB / -4.90275424742791E-01, -1.57647277946204E-03,
1 9.66195963140306E-05, -1.3591680268815E-07, -2.9815734254859E-07,
2 1.88824767559978E-08, 1.03685737657141E-09, -3.28660818434328E-10,
3 2.57091410832780E-11, 2.3235765300677E-12, -8.5752327904855E-13,
4 1.20340828049719E-13, 2.90907716770715E-15, -4.55656454560149E-16/
DATA BK1 / 2.43202846447449E+00, 2.57132009754685E+00,
1 1.02802341258618E+00, 3.4195817825872E-01, 8.4197862888284E-02,
2 1.93077282587962E-02, 3.92687837130335E-03, 6.83302689948045E-04,
3 1.14611403991141E-04, 1.741951788337086E-05, 2.41223629958355E-06,
4 3.2452591983273E-07, 4.03508788540183E-08, 4.70875059642298E-09,
5 5.3536743258589E-10, 5.70606721846334E-11, 5.80526363709933E-12,
6 5.76338988616388E-13, 5.42103834518071E-14, 4.91857330301677E-15/
DATA BK2 / 5.74830555784088E-01, -6.91648648376891E-03,
1 1.97460263052093E-03, -5.24043043868923E-04, 1.23965147239661E-04,
2 -2.27059514462173E-05, 2.2357555008526E-06, 4.15174955023899E-07,
3 -2.84985752198231E-07, 8.501871775435E-08, 1.70400828891326E-08,
4 2.25479746746899E-09, -1.09524166577443E-10, -3.41063845099711E-11,
5 1.11262893886662E-11, -1.75542944241734E-12, 1.36298606401767E-13,
6 8.76342105755664E-15, -4.64063099187041E-15, 7.78772758732960E-16/
DATA BK3 / 5.8677703506912E-01, 2.63672828349579E-03,
1 5.12303351473130E-05, 2.10229231564492E-06, 1.42217095113890E-07,
2 1.2852429591264E-08, 7.28556219407507E-10, -3.45236157301011E-10,
3 -2.11919115912724E-10, -8.5880389292376E-11, -8.14873160315074E-12,
4 3.03177845832183E-12, 1.73447220554115E-12, 1.67935548701554E-13,
5 -1.4982288806719E-13, -5.15470458953407E-14, 8.75741841857830E-15,
6 7.9673555352520E-15, -1.29586137861742E-16, -1.187879447520E-15/
DATA BK4 / 4.8544386705114E-01, -3.08525088408463E-03,
1 6.98748404837928E-05, -2.82757234179768E-06, 1.59553313064138E-07,
2 -1.12980692144601E-08, 9.47671515498754E-10, -9.08301736026423E-11,
3 9.70776206450724E-12, -1.13687527254574E-12, 1.43982917533415E-13,
4 -1.95211019558815E-14, 2.01056379909357E-15, -4.26916444775176E-16/
DATA BNP / 1.34918611457638E-01, -3.19314588205813E-01,
1 5.22061846276114E-02, 5.2888912170312E-02, -8.58100756077350E-03,
2 -2.99211002025555E-03, 4.21126741969759E-04, 8.73931820369273E-05,
3 1.06749163477533E-05, -1.56575097253492E-06, 1.68051151983999E-07,
4 1.06901103638691E-08, -1.81374004961922E-09, -1.66339134593739E-10,
5 1.42956335780810E-11, 1.10179811626595E-12, -8.60187724192263E-14,
6 -5.71248177285064E-15, 4.08414552853803E-16/
DATA BUN / 6.59041673525697E-02, -4.24905910566004E-01,
1 2.87209745195830E-01, 1.2978771099606E-01, -4.56354317590358E-02,
2 -1.02630175982540E-02, 2.50704671521101E-03, 3.78127183743483E-04,
3 -7.1128758384084E-05, -8.08651210688923E-06, 1.2387953127285E-06,
4 1.13096815867279E-07, -1.48234283178310E

```

C
D/ DBK1 / 2.95926143981893E+00, 3.8677456844E+00,
1 1.0441072356289E+00, 5.78070764125328E-01, 1.6301146817A, JE-01,
2 3.9204409961855E-02, 7.90964210433812E-03, 1.50640863167338E-03,
3 2.56651976920042E-04, 3.93826605867715E-05, 5.8109771463818E-06,
4 7.86981233754559E-07, 9.93272957325739E-08, 1.2142405575107E-08,
5 1.38526332697707E-09, 1.50190057886758E-10, 1.58271945457594E-11,
6 1.57531847695042E-12, 1.50774055398181E-13, 1.40594335806564E-14,
7 1.24942698777218E-15/

C
DATA DBK2 / 5.49756809432471E-01, 9.13556983276901E-03,
1-2.53635048605507E-03, 6.60423795342054E-04, -1.55217243135416E-04,
2 3.0090325448633E-05, -3.76454339487348E-06, -1.33291331611616E-07,
3 2.42587371049013E-07, -8.07861075240228E-08, 1.71092818861193E-08,
4-2.41087357570599E-09, 1.53910848162371E-10, 2.56465373190630E-11,
5-9.8858191165321E-12, 1.60877986412631E-12, -1.2095252471739E-13,
6-1.06578278410820E-14, 5.02478557067561E-15, -8.68986130335866E-16/

C
DATA DBK3 / 5.60598509354302E-01, -3.64870013248135E-03,
1-5.98147152307417E-05, -2.33611595253625E-06, -1.6457151621436E-07,
2-2.06333012920569E-08, -4.27745431573110E-09, -1.08494137799276E-09,
3-2.37207188872763E-10, -2.22132920864966E-11, 1.07238008032138E-11,
4 5.71954843245808E-12, 7.5110273777835E-13, -3.81912369483793E-13,
5-1.75870057119257E-13, 6.69641694419084E-15, 2.26866724792055E-14,
6 2.68988141356743E-15, -2.67133612307359E-15, -6.54121403165269E-16/

C
DATA DBK4 / 4.93072999188036E-01, 4.38335419803815E-03,
1-8.3741388246205E-05, 3.20268810484632E-06, -1.75661979548270E-07,
2 1.22269906524508E-08, -1.01381314386052E-09, 9.63639784237475E-11,
3-1.02344993379648E-11, 1.19284576554355E-12, -1.50443899103287E-13,
4 2.03299052379349E-14, -2.91890652008292E-15, 4.42322081975475E-16/

C
DATA DBJP / 1.1314087330745E-01, -2.08301511416328E-01,
1 1.69396341953138E-02, 2.90895212478621E-02, -3.41467131311549E-03,
2-1.46455339197417E-03, 1.63313272898517E-04, 3.9114532822162E-05,
3-3.96757190808115E-06, -8.51846913772395E-07, 5.98707495269280E-08,
4 7.44108654536549E-09, -6.21241056522832E-10, -6.1876801731326E-11,
5 4.7232348475324E-12, 3.91652459802532E-13, -2.74985937845226E-14,
6-1.95036497762750E-15, 1.26669643809444E-16/

C
DATA DBJN /-1.880912600688950E-02, -1.47798180826140E-01,
1 5.46075900433171E-01, 1.52146932663116E-01, -9.58260412266886E-02,
2-1.63102731696130E-02, 5.75364806680105E-03, 7.12145408252655E-04,
3-1.754521116846724E-04, -1.71063171685128E-05, 3.24435580531680E-06,
4 2.61190663932884E-07, -4.03026865912779E-08, -2.76435165853895E-09,
5 3.59687929062312E-10, 2.14953306456051E-11, -2.41849311903901E-12,
6-1.28068004920751E-13, 1.26939834401773E-14/

C
DATA DAA / 2.77571356944231E-01, -4.44212833419920E-03,
1 8.42328522190089E-05, 2.58040318418710E-06, -3.42389720217621E-07,
2 6.24286894709776E-09, 2.36377836844577E-09, -3.16991042656673E-10,
3 4.40995691658191E-12, 5.18674221093575E-12, -9.64874015137022E-13,
4 4.90190576608710E-14, 1.77253430678112E-14, -5.55950610442662E-15/

C
DATA DBB / 4.91627321104601E-01, 3.11164930427489E-03,
1 8.23140762854081E-05, -4.61769776172142E-06, -6.13158880534626E-08,
2 2.8279580465520E-08, -1.81959715372117E-09, -1.4475282642035E-10,
3 4.53724043420422E-11, -3.99655065847223E-12, -3.24089119830323E-13,
4 1.62098952568741E-13, -2.40785247974057E-14, 1.69384811284491E-16/

C
AX=ABS(X)
RX=SQRT(AX)
C=CON1*AX*RX
IF(X.LT.0.) GO TO 300
IF(C.GT.8.) GO TO 200
IF(C.GT.8.) GO TO 200

```

```

IF(X.GT.2.5) GO TO 150
T = (X-2.5)*.4
J=N1
F1=BK1(J)
F2=0.
DO 105 I=1,M1
  J=J-1
  TEMP1=F1
  F1=TT*F1-F2+BK1(J)
  F2=TEMP1
105 CONTINUE
  B1=TT*F1-F2+BK1(1)
  J=N1D
  F1=DK1(J)
  F2=0.
  DO 106 I=1,M1D
    J=J-1
    TEMP1=F1
    F1=TT*F1-F2+DK1(J)
    F2=TEMP1
106 CONTINUE
    DB1=TT*F1-F2+DK1(1)
    RETURN
150 CONTINUE
  RTRX=SQRT(RX)
  T=(X+X-CON2)*CON3
  TT=TT+T
  J=N1
  F1=BK2(J)
  F2=0.
  DO 155 I=1,M1
    J=J-1
    TEMP1=F1
    F1=TT*F1-F2+BK2(J)
    F2=TEMP1
155 CONTINUE
    B1=(TT*F1-F2+BK2(1))/RTRX
    EX=EXP(C)
    B1=B1*EX
    J=N2D
    F1=DK2(J)
    F2=0.
    DO 156 I=1,M2D
      J=J-1
      TEMP1=F1
      F1=TT*F1-F2+DK2(J)
      F2=TEMP1
156 CONTINUE
      DB1=(TT*F1-F2+DK2(1))/RTRX
      DB1=DB1*EX
      RETURN
C
200 CONTINUE
  RTRX=SQRT(RX)
  T=16./C-1.
  TT=TT+T
  J=N1
  F1=BK3(J)
  F2=0.
  DO 205 I=1,M1
    J=J-1
    TEMP1=F1
    F1=TT*F1-F2+BK3(J)
    F2=TEMP1
205 CONTINUE

```

```

S1=7*F1-F2+BK3(1)
J=J-1
F1=BK3(J)
F2=0.
DO 206 I=1,M2D
  J=J-1
  TEMP1=F1
  F1=TT*F1-F2+BK3(J)
  F2=TEMP1
206 CONTINUE
D1=7*F1-F2+BK3(1)
TC=C+C
EX=EXP(C)
IF(TC.GT.35.) GO TO 214
T=10./C-1.
TT=TT+T
J=N3
F1=BK4(J)
F2=0.
DO 210 I=1,M3
  J=J-1
  TEMP1=F1
  F1=TT*F1-F2+BK4(J)
  F2=TEMP1
210 CONTINUE
S2=7*F1-F2+BK4(1)
B1=(S1+EXP(-TC)*S2)/RTRX
B1=B1*EX
J=N4D
F1=BK4(J)
F2=0.
DO 211 I=1,M4D
  J=J-1
  TEMP1=F1
  F1=TT*F1-F2+BK4(J)
  F2=TEMP1
211 CONTINUE
D2=7*F1-F2+BK4(1)
DB1=RTRX*(D1+EXP(-TC)*D2)
DB1=DB1*EX
RETURN
214 B1=EX*S1/RTRX
DB1=EX*RTRX*D1
RETURN
C
300 CONTINUE
IF(C.GT.5.) GO TO 350
T=4*C-1.
TT=TT+T
J=N2
F1=BJP(J)
E1=BJN(J)
F2=0.
E2=0.
DO 305 I=1,M2
  J=J-1
  TEMP1=F1
  TEMP2=E1
  F1=TT*F1-F2+BJP(J)
  E1=TT*E1-E2+BJN(J)
  F2=TEMP1
  E2=TEMP2
305 CONTINUE
B1=(T*E1-E2+BJN(1))-AX*(T*F1-F2+BJP(1))
J=N3D
F1=BJP(J)

```

```

E1=DBUN(J)
F2
E2
DO 306 I=1,M3D
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2*DBUN(J)
E1=TT*E1-E2*DBUN(J)
F2=TEMP1
E2=TEMP2
306 CONTINUE
DBI=X*X*(T*F1-F2*DBUN(1))+(T*E1-E2*DBUN(1))
RETURN

C
350 CONTINUE
RTRX=SQRT(RX)
T=10./C-1.
TT=T+T
J=N3
F1=AA(J)
E1=BB(J)
F2=0.
E2=0.
DO 310 I=1,M3
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2*AA(J)
E1=TT*E1-E2*BB(J)
F2=TEMP1
E2=TEMP2
310 CONTINUE
TEMP1=T*F1-F2*AA(1)
TEMP2=T*E1-E2*BB(1)
CV=C-SP112
BI=(TEMP1*COS(CV)+TEMP2*SIN(CV))/RTRX
J=N4D
F1=DAA(J)
E1=DBB(J)
F2=0.
E2=0.
DO 311 I=1,M4D
J=J-1
TEMP1=F1
TEMP2=E1
F1=TT*F1-F2*DAA(J)
E1=TT*E1-E2*DBB(J)
F2=TEMP1
E2=TEMP2
311 CONTINUE
TEMP1=T*F1-F2*DAA(1)
TEMP2=T*E1-E2*DBB(1)
CV=C-SP112
DBI=(TEMP1*COS(CV)-TEMP2*SIN(CV))/RTRX
RETURN
END
SUBROUTINE BESKN (X,MU,KODE,N,Y,NZ)
C
C WRITTEN BY D.E. AMOS AND S.L. DANIEL, FEBRUARY, 1974.
C
C REFERENCE SAND-75-0151
C
C ABSTRACT
C BESKN IMPLEMENTS FORWARD RECURSION ON THE THREE TERM
C RECURSION RELATION FOR A SEQUENCE OF INTEGER ORDER BESSEL

```



```

NN=MINO(2,ND)
FY IUD
FN JD=ND-1
FNN=FN
IF(FN.LT.2.) GO TO 300

C OVERFLOW TEST (LEADING EXPONENTIAL OF ASYMPTOTIC EXPANSION)
C FOR THE LAST ORDER, NU=N-1,GE,NULIM
C
ZN=X/FN
RTZ=SORT(1.+ZN*ZN)
GLN=ALOG((1.+RTZ)/ZN)
T=RTZ*(1.-ETX)+ETX/(ZN+RTZ)
CN=FN*(T-GLN)
IF(CN.GT.ELIM) GO TO 94
IF(NUD.LT.NULIM(NN)) GO TO 200
IF(NN.EQ.1) GO TO 10
S CONTINUE

C UNDERFLOW TEST (LEADING EXPONENTIAL OF ASYMPTOTIC EXPANSION)
C FOR THE FIRST ORDER, NU,GE,NULIM
C
FN=FNU
ZN=X/FN
RTZ=SORT(1.+ZN*ZN)
GLN=ALOG((1.+RTZ)/ZN)
T=RTZ*(1.-ETX)+ETX/(ZN+RTZ)
CN=FN*(T-GLN)
10 IF(CN.LT.-ELIM) GO TO 95
20 CONTINUE

C ASYMPTOTIC EXPANSION FOR ORDERS NU AND NU+1,GE,NULIM
C
CALL ASKBES(KODE,FNU,NM,X,RTZ,CN,Y)
GO TO (96,27). NM
27 TRX=2./X
TM=TRX*(FNU+1.)
GO TO 250

C 200 CONTINUE
IF(KODE.EQ.2) GO TO 201

C UNDERFLOW TEST (LEADING EXPONENTIAL OF ASYMPTOTIC EXPANSION IN X)
C FOR ORDER 0
C
204 IF(X.GT.ELIM) GO TO 95
201 S1=BESKO(X,0,KODE,MZ)
S2=BESKO(X,1,KODE,MZ)
TRX=2./X
TM=TRX
HERE NU=0 OR 1 IMPLIES N,GE.2. OR NU,GE.2 IMPLIES N,GE.1
IN=NUD-1
IF(IN) 220,211,203
FORWARD RECUR FROM 0 TO NU TO GET Y(1)
203 DO 210 I=1,IN
S=S2
S2=TM+S2+S1
S1=S
TM=TM+TRX
210 CONTINUE
211 IF(ND.EQ.1) GO TO 251
S=S2
S2=TM+S2+S1
S1=S
TM=TM+TRX

```

```

220 CONTINUE
C 221 F1 3RD RECUR FROM NU+2 TO NU+N-1
Y(1)=S1
Y(2)=S2
250 CONTINUE
IF(ND.EQ.2) GO TO 96
NM2=ND-2
DO 230 I=1,NM2
Y(I+2)=TM*Y(I+1)+Y(I)
TMMATRIX
230 CONTINUE
GO TO 96
251 Y(1)=S2
GO TO 96
C 300 CONTINUE
C FN.LT.2 IMPLIES THE CASES NU=0, N=1 OR 2
NU=1, N=1
GO TO (309,310),KODE
C UNDERFLOW TEST FOR KODE=1
309 IF(X.GT.ELIM) GO TO 95
310 CONTINUE
NUM1=NUD-1
DO 315 I=1,ND
J=NUM1+I
ANS=BESK01(X,J,KODE,M2)
Y(I)=ANS
315 CONTINUE
GO TO 96
C UPDATE PARAMETERS ON UNDERFLOW
95 NUD=NUD+1
NO=NO-1
IF(ND.EQ.0) GO TO 96
NM=MIN0(2,ND)
FNU=NUD
IF(FNN.LT.2.) GO TO 95
IF(MUD.LT.NULIM(NN)) GO TO 200
GO TO 5
96 NZ=N-ND
IF(NZ.EQ.0) RETURN
IF(ND.EQ.0) GO TO 98
DO 97 I=1,ND
J=N-I+1
K=ND-I+1
Y(J)=Y(K)
97 CONTINUE
98 DO 99 I=1,NZ
Y(I)=0.
99 CONTINUE
RETURN
C 90 CONTINUE
CALL ERRCHK(43,43HIN BESKN, SCALING OPTION, KODE, NOT 1 OR 2.)
RETURN
91 CONTINUE
CALL ERRCHK(36,36HIN BESKN, ORDER, NU, LESS THAN ZERO.)
RETURN
92 CONTINUE

```



```

CA)  ERRCHK(39.39*IN BESKN, X LESS THAN OR EQUAL TO ZERO.)
      N
93  CONTINUE
      CALL ERRCHK(26.26*IN BESKN, N LESS THAN ONE.)
      RETURN
94  CONTINUE
      CALL ERRCHK(53.53*IN BESKN, OVERFLOW, NU OR N TOO LARGE OR X TOO S
      1MALL.)
      RETURN
      END
SUBROUTINE ASKBES(KODE,FNU,IN,X,RA,ARG,Y)
CDC 6600 ROUTINE
JANUARY 1974

```

ASKBES COMPUTES K BESSEL FUNCTIONS
 $K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$
 OR SCALED BESSEL FUNCTIONS
 $\text{EXP}(X)*K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$
 FOR $X.GT.0.$, $FNU.GE.32.$, AND $IN=1$ OR 2

INPUT

KODE - A PARAMETER TO INDICATE THE SCALING OPTION
 KODE=1 RETURNS $Y(I)=K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$
 KODE=2 RETURNS $Y(I)=\text{EXP}(X)*K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$
 FNU - ORDER OF FIRST BESSEL FUNCTION
 IN - NUMBER OF K FUNCTIONS DESIRED, $IN=1$ OR 2
 X - ARGUMENT, $X.GT.0.$
 RA - $\text{SORT}(1+Z/2)$, $Z=X/FNU$
 ARG - ARGUMENT OF THE LEADING EXPONENTIAL

OUTPUT

Y - A VECTOR WHOSE FIRST IN COMPONENTS CONTAIN THE SEQUENCE
 $Y(I)=K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$ OR
 $Y(I)=\text{EXP}(X)*K/\text{SUB}(FNU+1-1)/(X)$, $I=1, IN$
 DEPENDING ON KODE

WRITTEN BY
 D. E. AMOS
 S. L. DANIEL

ABSTRACT
 ASKBES IMPLEMENTS THE UNIFORM ASYMPTOTIC EXPANSION OF
 $K/\text{SUB}(FNU)/(X)$ FOR $FNU.GE.32$ AND REAL $X.GT.0.$ THE FORM, EXCEPT
 FOR AN ALTERNATION OF SIGN IN THE EXPANSION, IS IDENTICAL TO
 THAT IMPLEMENTED IN BESI.

DIMENSION Y(1)
 DIMENSION C(11,10), C1(11,8), C2(11,2)
 EQUIVALENCE (C(1,1), C1(1,1))
 EQUIVALENCE (C(1,9), C2(1,1))
 DATA TOL/1.E-15/
 DATA RTPIO2/1.25331413731550/

DATA C1 9(0.) /-2.08333333333333E-01, 1.25000000000000E-01,
 1 7.03125000000000E-02, 3.34201388888889E-01, -4.01041666666667E-01,
 2 1.84646267361111E+00, -8.91210937500000E-01, 7.32421875000000E-02,
 3 7(0.) /4.66958442342825E+00, -1.12070026162230E+01,
 4 8.7891235315625E+00, -2.36408691406250E+00, 1.12152099609375E-01,
 5 6(0.) /-2.82120725582002E+01, 8.46362176746007E+01,
 6 7-9.18182415432400E+01, 4.2534987453885E+01, -7.36879435947963E+00,
 8 2.27108001708984E-01, 5(0.) /2.12570130039217E+02,

```

9-7. 45252489141182E+02, 1.05998045282800E+03, -6.995796277376133E+02,
A 2( 190511744312E+02, -2.6491430489516E+01, 5.725014209747 -01,
4(0.) -1.91945766231841E+03, 8.061722181737 +03,
C-1.35865500664341E+04, 1.16553933368645E+04, -5.30564697861340E+03,
D 1.20090291321635E+03, -1.08090919788395E+02, 1.7272750258446E+00,
E 3(0.) 2.02042913309661E+04, -9.69805983886375E+04,
F 1.92547001232532E+05, -2.03400177280416E+05, 1.22200464983017E+05,
G-4.11926549688976E+04, 7.10951430248936E+03, -4.93915304773088E+02,
H 6.07404200127348E+00, 2(0.) /

```

C

```

DATA C2
-2.42919187900551E+05, 1.31176361466298E+06,
1-2.99801591853811E+06, 3.76327129765640E+06, -2.81356322658653E+06,
2 1.2683652732162E+06, -3.31645172484564E+05, 4.52187689813627E+04,
3-2.49983048181121E+03, 2.4380529695561E+01, 1(0.)
4 3.28446985307204E+06, -1.97068191184322E+07, 5.09526024926646E+07,
5-7.41051482115327E+07, 6.63445122747290E+07, -3.75671766607634E+07,
6 1.32887671664218E+07, -2.78561812808645E+06, 3.08186404612662E+05,
7-1.38860897537170E+04, 1.10017140269247E+02/

```

C

```

FN=FNH
ETX=KODE-1
DO 200 JN=1, IN
IF(JN.EQ.1) GC TO 148
FN=FN+1.
Z=X/FN
RA=SQRT(1.+Z*Z)
GLN=ALOG((1.+RA)/Z)
T=RA*(1.-ETX)+ETX/(Z+RA)
ARG=-FN*(T-GLN)
148 COEF=EXP(ARG)
T=1./RA
T2=T*T
T3=T*T
S2=1.
AP=1.
DO 140 K=1,10
KPI=K+1
S1=C(1,K)
DO 135 J=2,KPI
S1=S1*T2+C(J,K)
135 CONTINUE
AP=AP*T
AK=AP*S1
S2=S2*AK
IF(ABS(AK).LT.TOL) GO TO 145
140 CONTINUE
145 CONTINUE
Y(JN)=SQRT(T)*RTPID2*COEF*S2
200 CONTINUE
RETURN
END
FUNCTION GAMLN(X)

```

GMLN 1
GMLN 2
GMLN 4

WRITTEN BY D. E. AMOS, SEPTEMBER, 1977.

REFERENCES

SAND-77-1518

COMPUTER APPROXIMATIONS BY J.F. MART, ET AL., SIAM SERIES IN
APPLIED MATHEMATICS, WILEY, 1968, P.135-136.

NBS HANDBOOK OF MATHEMATICAL FUNCTIONS, AMS 55, BY
M. ABRAMOWITZ AND I.A. STEGUN, DECEMBER, 1955, P.257.

GMLN 6
GMLN 7
GMLN 8

ABSTRACT

GAMLN COMPUTES THE NATURAL LOG OF THE GAMMA FUNCTION FOR


```

6 2 0564940971863E+02, 2.64921649798553E+02, 2.69291097651070E+02, GMLN 61
7 1 673124285694E+02, 2.78067573440366E+02, 2.82474292687E+02, GMLN 62
8 2 6893133295427E+02, 2.91323950094270E+02, 2.957666013507E+02, GMLN 63
9 3 00220948847014E+02, 3.0468685676569E+02, 3.09164193580147E+02, GMLN 64
A 3.1265282949879E+02, 3.18152639620209E+02, 3.22663499128726E+02, GMLN 65
B 3.2718528770377E+02, 3.31717887196928E+02, 3.36281181979198E+02, GMLN 66
C 3.4081505887079E+02, 3.45379407062267E+02, 3.49954118040770E+02, GMLN 67
D 3.5453988519441E+02, 3.59134205369575E+02, GMLN 68
C IF(X) 90.90.5 GMLN 69
5 NDX=X
T=X-FLOAT(NDX)
IF(T.EQ.0.0) GO TO 51
DX=X-LIM1-X
IF(DX.LT.0.0) GO TO 40
C RATIONAL CHEBYSHEV APPROXIMATION ON 2.LT.X.LT.3 FOR GAMMA(X)
C
C NXM=NDX-2
PX=PCOE(1)
DO 10 I=2,9
10 PX=T*PX+PCOE(I)
QX=QCOE(1)
DO 15 I=2,4
15 QX=T*QX+QCOE(I)
DGAM=PX/QX
IF(NXM.GT.0) GO TO 22
IF(NXM.EQ.0) GO TO 25
C BACKWARD RECURSION FOR 0.LT.X.LT.2
C
C DGAM=DGAM/(1.+T)
IF(NXM.EQ.-1) GO TO 25
DGAM=DGAM/T
GMLN=ALOG(DGAM)
RETURN
C FORWARD RECURSION FOR 3.LT.X.LT.8
C
C 22 XX=2.+T
DO 24 I=1,NXM
DGAM=DGAM*XX
24 XX=XX+1.
25 GMLN=ALOG(DGAM)
RETURN
C X.GT.XLIM1
C
C 40 RX=1./X
RX=RX*RX
IF((X-XLIM2).LT.0.) GO TO 41
PX=Q(1)*RX+Q(2)
GMLN=PX*RX+(X-.5)*ALOG(X)-X*RTWPIL
RETURN
C X.LT.XLIM2
C
C 41 PX=P(1)
SUM=(X-.5)*ALOG(X)-X
DO 42 I=2,5
PX=PX*RX+P(I)
42 CONTINUE
GMLN=PX*RX+SUM+RTWPIL
RETURN
C TABLE LOOK UP FOR INTEGER ARGUMENTS LESS THAN OR EQUAL 100.

```

```

C 51 IF (X.GT.100) GO 10,40
    GAM=N=GLN(NDX)
    RETURN
90 CALL ERRCHK(49,49HIN GAMLN, ARGUMENT IS LESS THAN OR EQUAL TO ZERO
    1.)
    RETURN
    END
    GMLN 119
    GMLN 120
    GMLN 121
    GMLN 122
    GMLN 123
    GMLN 124
    GMLN 125

```

Program Listing

- GRATING -

Subroutines: PDECOL, PLOTN

(Reference: Vol. I, Section III)

WHICH OUTPUT INFORMATION IS DESIRED (SOLUTION ARRAYS, FIELDS, ETC.) ARE
 OR EACH CARD-IMAGE GIVES A TIME INCREMENT AND THE NUMBER (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12) OF
 CONSECUTIVE INCREMENTS OF THIS SIZE. TOGETHER THESE INCREMENTS CON-
 STITUTE A SET OF STEPS. AT THE SUCCESSFUL COMPLETION OF SUCH A SET OF
 STEPS, CONTROL RETURNS TO STATEMENT 3 FOR DEFINITION OF A NEW STEP OR A
 NEW SET OF STEPS. (INFORMATION ON PLOTTING AND BEGINNING THE ERASE MODE
 IS ALSO READ HERE; SEE BELOW.) A BLANK CARD HERE IS THE NORMAL MEANS
 OF TERMINATING PROGRAM EXECUTION.

A TYPICAL DATA-SET MIGHT BE (SEE COMMENTS AT STATEMENTS 1, 2, AND 3):

```

C----- 100 12 5 4 0 4 2
C----- 1.00 9 0.04 0.02 1.00 NOTE DEFAULT VALUES OF G1, G2
C----- .10E-08 10 PLOT IS READ AS ZERO
C----- .10E-07 9 4 PLOT RHO, E AT T = 1.0E-06
C----- .10E-06 9
C----- .10E-05 9
C----- .10E-04 9
C----- .10E-03 9
C----- .10E-02 9
C----- .10E-01 9
C----- .10E+00 9
C----- .10E+01 9 4ERASE
C----- .10E+02 10
C----- .10E-01 9
C----- .10E+00 9 4
C----- .10E+01 9
C----- .10E+02 9
C----- .10E+03 9
C----- .10E+04 9
C----- .10E+05 9
C----- .10E+06 9
C----- .10E+07 9
C----- .10E+08 9
C----- .10E+09 9
C----- .10E+10 9
C----- .10E+11 9
C----- .10E+12 9
  
```

SUBROUTINES CONTAINED ON THE MAIN SOURCE FILE:

```

C BLOCK DATA
C SUBR. STACK
C SUBR. POUT
C SUBR. CHECK
C SUBR. F
C FUNCTION LGT
C SUBR. BNDRY
C SUBR. UNIT
C SUBR. SCFLD
C SUBR. EFELD
C SUBR. ESTOR
C SUBR. RERR
C SUBR. INT
C SUBR. ALI
C SUBR. PEAK
  
```

SUBPROGRAMS CONTAINED IN THE PACKAGE PDECOL ARE DESCRIBED IN THAT LISTING
 SUBROUTINE PLOTN IS DESCRIBED IN THE PLOTN LISTING

NOTES ON PARAMETERS USED FOR TIME AND TIME-INCREMENTS:

T-----CURRENT WORKING TIME, CHOSEN BY THE PACKAGE (TO ADVANCE THE
 TIME-INTEGRATION PROCESS AS QUICKLY AS POSSIBLE CONSISTENT
 WITH MAINTAINING THE REQUIRED ACCURACY). WORKING-TIME STEPS
 ARE COUNTED BY VARIABLE NSTEP.

TOUT---OUTPUT TIME, SPECIFIED BY USER, AT WHICH SOLUTIONS ARE DE-
 Sired. OUTPUT-TIME STEPS ARE COUNTED BY VARIABLE KNT.

DELT---SIZE OF OUTPUT-TIME STEPS (READ IN AT STATEMENT 3)
 DTUSED---SIZE OF WORKING-TIME STEPS (DETERMINED BY THE PACKAGE AND
 PRINTED AS "DT" UNDER "TIME-STEP-SEQUENCE" ON TAPES).
 DT1---INITIAL WORKING-TIME STEP USED ONLY ON FIRST CALL TO PDECOL
 DT2---SIZE OF FIRST THREE STEPS AFTER BEGINNING OF ERASE MODE.
 WORKING TIME COINCIDES WITH OUTPUT TIME FOR THESE STEPS.

10 ---INITIAL TIME (ZERO)
 11 ---ERASE-TIME: TIME MEASURED FROM TO WHEN ERASE MOD .EGINS
 12 ---CURRENT OUTPUT TIME, MEASURED FROM TO OR TER
 13 ---VALUE OF T AT PREVIOUS WORKING-TIME STEP

NOTES ON THE DETERMINATION OF THE SPACE-CHARGE FIELD:

THE ELECTRIC FIELD CONTAINS AN APPLIED-FIELD COMPONENT (READ IN AS B) AND A SPACE-CHARGE COMPONENT (DETERMINED AT EACH STEP BY INTEGRATING RHO, THE SPACE-CHARGE, FROM MINUS INFINITY TO X). SINCE THE FIELD APPEARS AS COEFFICIENTS IN THE PARTIAL DIFFERENTIAL EQUATIONS (PDES), THE EQUATIONS ARE COUPLED NONLINEAR INTEGRO-DIFFERENTIAL EQUATIONS. TO MAKE THESE TRACTABLE THE INTEGRAL IS REPLACED BY AN "ANTICIPATED" VALUE OF THE SPACE-CHARGE FIELD. IT IS VERY IMPORTANT THAT THIS "ANTICIPATED" VALUE NOT BE GREATLY IN ERROR.

BOTH THE "ANTICIPATED" AND THE "ACTUAL" (SEE BELOW) VALUES OF THE SPACE-CHARGE FIELD ARRAYS ARE EVALUATED AT THE BREAK-POINTS, XBKPT. WHATEVER INTERMEDIATE SPATIAL POINTS ARE REQUIRED ARE OBTAINED BY INTERPOLATION.

AT THE END OF EACH WORKING-TIME STEP AND (IF T.NE.TOUT, AS IS USUALLY THE CASE) ALSO AT EACH OUTPUT-TIME STEP, THE "ACTUAL" FIELD ARRAY IS DETERMINED. THIS IS DONE IN SUBROUTINE SCFLD BY NUMERICAL INTEGRATION OF THE SOLUTION ARRAYS. (RHO IS PROPORTIONAL TO THE DIFFERENCE OF THE TWO SOLUTIONS, V-U.)

SUPPOSE THE MOST RECENT STEP TO BE COMPLETED IS THE NTH ONE, CORRESPONDING TO TIME T. THE "ACTUAL" FIELD FOR THIS STEP (AS WELL AS ALL PREVIOUS ONES) HAS THEREFORE BEEN DETERMINED FROM THE SOLUTION ARRAYS. AT THE BEGINNING OF STEP N+1, T IS AUGMENTED TO T+DT, A NEW "ANTICIPATED" VALUE OF THE FIELD ARRAY IS COMPUTED BY TAKING THE PREVIOUS THREE "ACTUAL" FIELD ARRAYS AND EXTRAPOLATING TO THE NEW VALUE OF T. (THESE ARRAYS, STORED AS ES, CORRESPOND TO STEPS N-2, N-1, AND N; THE THREE TIMES ARE STORED IN ARRAY TE.)

WHEN STEP N+1 IS COMPLETE A NEW "ACTUAL" FIELD ARRAY IS COMPUTED. AN ERROR PARAMETER, DIS, GIVES THE AVG ABSOLUTE ERROR IN THE "ANTICIPATED" FIELD FOR STEP N+1 BY COMPARING THE ARRAY E TO THIS NEW "ACTUAL" FIELD ARRAY. (DIS IS ACTUALLY COMPUTED AT THE BEGINNING OF STEP N+2.) SINCE THE "ACTUAL" FIELD IS DETERMINED AFTER A CALCULATION WHICH USED THE "ANTICIPATED" FIELD, IT IS IMPORTANT TO MINIMIZE THIS ERROR. THIS CAN BE DONE (AT A COST IN CP TIME) BY USING SMALL VALUES OF EPS, AND/OR BY FORCING THE PROGRAM TO TAKE SMALL WORK-TIME STEPS. (SEE PDECOL)

VARIABLES WITH SINGLE ASSIGNED PURPOSES IN THE MAIN PROGRAM:

A DIFFUSION LENGTH (READ IN AT STATEMENT 2)
 B DRIFT LENGTH (READ IN AT STATEMENT 2)
 DIS DISCREPANCY BETWEEN "ANTICIPATED" FIELD AND "ACTUAL" FIELD
 DT DTZ, ETC---SEE COMMENTS ABOVE
 E ARRAY FOR STORING VALUES OF "ANTICIPATED" SPACE-CHARGE FIELD
 EP ERROR PARAMETER FOR SUBROUTINE PEAK
 EPS TIME-DISCRETIZATION ERROR PARAMETER (SEE COMMENTS IN PDECOL)
 ER CHARACTER CONSTANT NEEDED TO INITIATE ERASE MODE
 ERASE CHARACTER VARIABLE NEEDED TO INITIATE ERASE MODE
 ES ARRAY TO STORE VALUES OF "ACTUAL" SPACE-CHARGE FIELD
 G1 RECOMBINATION FREQUENCY (READ IN AT STATEMENT 2)
 G2 EXCITATION FREQUENCY (READ IN AT STATEMENT 2)
 ICNT # OF TIMES FUNCTION F IS CALLED
 INDEX CONTROL CODE FOR PDECOL (SEE COMMENTS IN PDECOL)
 INDX VALUE USED FOR INDEX FOR MOST OF CALCULATION (READ IN AT STATEMENT 1)

[illegible]

[illegible]

[illegible]


```

CALL PEAK(SPACE(MBEG),XBKPT(MBEG),MMUM,YMIN,DY,EP,1)
VT YMAX = YMIN
WH .EIG.(105)-AT.NSTEP.TIME1.TCUR,U(1,IC,1).U(2,IC,1).ES(IC, >),
VT XMAX,YMAX,YMIN
IF(ITER.NE.K)GO TO 365
DO 350 I = 1,NPTS
350 SPACE(I) = U(1,I,1)
CALL POUT(4,TCUR,SPACE,TER,1,NPTS,NO,PRT(1))
IF(B.GT.0.0)GO TO 359
VT = 10.0/G1
IF(A.GT.0.0.OR.TOUT.GT.VT)GO TO 359
DO 355 J = 1,NPTS
VT = XBKPT(J)
CALL CHECK(VT,TOUT,YMAX)
355 SPACE(J) = SPACE(I) - YMAX
CALLS POUT(4,TOUT,SPACE,TER,1,NPTS,NO,PRT(13))
359 DO 360 I = 1,NPTS
360 CALL POUT(4,TCUR,SPACE,TER,1,NPTS,NO,PRT(3))
CALL POUT(4,TCUR,SPACE,TER,1,NPTS,NO,PRT(11))
365 DO 370 I = 1,NPTS
370 SPACE(I) = U(2,I,1) - U(1,I,1)
CALL POUT(4,TCUR,SPACE,TER,1,NPTS,NO,PRT(11))
IF(PLC.GT.0.AND.K.EQ.ITER)CALL STACK(SPACE,WORK,M,TCUR,PRT(1)),
1,PLC,7)
DO 380 I = 1,NPTS
380 SPACE(I) = ES(1,ID)
CALL POUT(4,TCUR,SPACE,TER,1,NPTS,NO,PRT(5))
IF(PLC.GT.0.AND.K.EQ.ITER)CALL STACK(SPACE,WORK,M,TCUR,PRT(5)),
12,PLC,7)
C C RESET INDEX
C C INDEX = INDX
C C IF(K.EQ.(ITER-1)).AND.ERASE.EQ.ER)INDEX = 2
C C
C C 400 CONTINUE
C C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C C RESET ERASE MODE IF DESIRED. STRATEGY IS TO SET INDEX = 3 AND
C C ADVANCE THREE SMALL STEPS FROM THE FINAL OUTPUT TIME OF THE WRITE
C C MODE (WHICH WILL HAVE BEEN REACHED EXACTLY). THEN THE PROGRAM IS
C C ALLOWED TO TAKE OVER AND CHOOSE ITS OWN TIME STEPS, (UNLESS INDX
C C HAD BEEN SET EQUAL TO 2 AT THE BEGINNING).
C C
C C IF(ERASE.NE.ER)GO TO 3
C C DELT = D*TZ
C C DT = DTZ
C C INDEX = 3
C C ITER = 1
C C IF(KNT-3)399,480,500
C C
C C 480 ERASE = 5H
C C INDEX = INDX
C C TOUT = TER
C C ER = 5H BOMB
C C GO TO 3
C C
C C INITIALIZE CERTAIN ARRAYS AND PARAMETERS FOR THE ERASE MODE
C C
C C 500 M = 0.0
C C PLC = 0
C C TER = TOUT
C C KNT = 0
C C DIS = 0.0

```



```

C      SUBROUTINE TO PREPARE FOR PLOTTING THE SPACE-CHARGE OR THE
C      SPACE-CHARGE FIELD. STACK DETERMINES XPLOT, THE ARRAY OF VALUES
C      OF X, AND THE INTERFERENCE PATTERN AT THESE VALUES OF X. THEN IT
C      CALLS ROUTINES TO DETERMINE RHO AND E (UNLESS PLC = 1, IN WHICH
C      CASE THEY ARE DETERMINED IN MAIN) AND STORES THEM IN SPACE. LAST
C      OF ALL, IT SCALES THE VARIABLES SO THAT THE INTERFERENCE PATTERN
C      FILLS THE TOP 20 PERCENT OF THE PLOT REGION, AND CALLS PLOTN.
C
C      INTEGER PLC, PLL
C      REAL M, LGT
C      DIMENSION XB-PT(201), E(201), ES(201,3), TE(3)
C      DIMENSION PLVAR(201,4), STORE(201), XPLOT(201), SPACE(201), WRT(2)
C      DIMENSION UL(2,201), SCICH(4), WORK(1)
C      COMMON/EFIELD/XBKPT,E,ES,TE,TPRV,TER,DIS,NPTS,INTR,ICNT
C      EQUIVALENCE (PLVAR(1),U(1))
C
C      INITIALIZATION OF PARAMETERS. THE FIRST CALL SHOULD OCCUR WITH
C      PLC = 0. ON THIS CALL (ONLY) T, WRT(1), AND WRT(2) PROVIDE IN-
C      FORMATION ON WHERE TO BEGIN ANY EXPANDED PLOTS---THAT IS, WHAT
C      VALUE XMIN SHOULD HAVE.
C
C      IF (PLC.GT.0) GO TO 200
C      PLL = 0
C      ER = M
C      XMIN = WRT(2) + 0.8 + T + WRT(1)
C      XMIN = -XMIN
C      DO 180 I = 2,NPTS
C      IF (XMIN.LT.XBKPT(I)) GO TO 190
C      180 CONTINUE
C      190 XMIN = XBKPT(I-1)
C      RETURN
C
C      SECTION TO SET UP THE ARRAY, XPLOT, AND CALCULATE LGT(X)
C
C      200 IF (PLC.EQ.PLL) GO TO 300
C      PLL = PLC
C      IF (PLC.GT.1) GO TO 220
C      DO 210 I = 1,NPTS
C      X = XBKPT(I)
C      STORE(I) = LGT(X)
C      210 XPLOT(I) = X
C      MAX = NPTS
C      GO TO 350
C
C      PLC > 1: SET MAX AND XMAX SO THAT THE RANGE IS XMIN-ABS(XMIN) OR
C      XMIN-N (WHERE N IS A NONNEGATIVE INTEGER). CHOOSING THE GREATEST
C      SUCH RANGE THAT FITS INTO 161 PRINT POSITIONS. OTHERWISE PLOT 161
C      POINTS STARTING AT XMIN
C
C      220 IF (PLC.GT.10) PLC = 10
C      DX = (XBKPT(2) - XBKPT(1))/PLC
C      XMAX = ABS(XMIN)
C      XMAX = (XMAX - XMIN)/DX + 1.0 + 1.0E-10
C      IF (MAX.LE.161) GO TO 230
C      I = XMAX + 1.0 + 1.0E-10
C      XMAX = I
C      DO 225 J = 1,I
C      MAX = (XMAX - XMIN)/DX + 1.0 + 1.0E-10
C      IF (MAX.LE.161) GO TO 230
C      225 XMAX = XMAX - 1.0
C      MAX = 161
C      XMAX = XMIN + (MAX - 1)*DX
C      230 CONTINUE
C      DO 240 I = 1,MAX
C      X = XMIN + (I-1)*DX

```



```

C      BH (WRITE)
C      IF (EQ.0.0)MM = BH
C      IF (TER.GT.0.0)MM = BH (ERASE)
C      IF (NO.GT.10)NO = 10
C      J = 1
C      NL = (MAX - MIN + 1)/NO
C      IF ((NO-NL).EQ.(MAX-MIN+1))J = 0
C      NL = NL + J
C      WRITE(FOOT,102)WRT(1),WRT(2)
C      WRITE(FOOT,101)T,MM
C      DO 200 I = 1,NL
C      K = MIN + NO*(I-1)
C      L = K + NO - 1
C      IF (L.GT.MAX)L = MAX
C      IF (NO.LT.10)GO TO 199
C      IF ((L-K).LT.9)GO TO 199
C      WRITE(FOOT,100)K,(X(J),J=K,L),L
C      GO TO 200
C      199 WRITE(FOOT,100)K,(X(J),J=K,L)
C      200 CONTINUE
C      RETURN
C      END
C      SUBROUTINE CHECK(X,T,U)
C      SUBROUTINE TO CALCULATE THE VALUE OF U, AT A SINGLE POINT X AND
C      A TIME T, IN THE ABSENCE OF DRIFT AND DIFFUSION
C      REAL LGT
C      COMMON/A/DUM(3),G1,G2,R,DVM(S)
C      F = LGT(X)
C      A = R*G2
C      B = G1 + G2*F
C      C = -F*G1
C      ARG = SQRT(B*B - 4.0*A*C)
C      AL = B/ARG
C      CP = 2.0*A*AL/B
C      CPP = CP/(1.0 + AL)
C      CP = CP/(1.0 - AL)
C      F = EXP(-ARG*T)
C      U = (1.0 - F)/(CP + CPP*F)
C      RETURN
C      END
C      SUBROUTINE F(T,X,U,UX,UX,FVAL,NPDE)
C      DIMENSION U(NPDE),UX(NPDE),UX(NPDE),FVAL(NPDE)
C      REAL LGT
C      COMMON/A/PI,A,B,G1,G2,R,DUM(S)
C      THESE (FVAL) ARE THE RIGHT-HAND SIDES OF THE PDES TO BE EVALUATED
C      CALL EFIED(X,T,XZ)
C      SUM = LGT(X)*(G1 - G2*U(2)) - G1*U(1) - R*G2*U(1)*U(2)
C      SVM = (G1*B + XZ)*UX(1) + U(2)*U(1) - U(1)*U(1)
C      FVAL(1) = A*A*G1*UX(1) + SVM + SUM
C      FVAL(2) = SUM
C      RETURN
C      END
C      REAL FUNCTION LGT(X)
C      FUNCTION TO ESTABLISH LIGHT-INTENSITY FROM INTERFERENCE PATTERN
C      REAL M
C      COMMON/A/PI,DUM(5),M,XB,S1,S2,PH

```

```

IF(ABS(X).LT.XS)GO TO 200
R1 = 11--3
S1 = S1
Y = -XB
IF(X.LT.0.0)GO TO 100
S = S2
Y = XB
100 ARG = R*(X-Y)**2
Y = 0.0
IF(ARG.LT.800.)Y = EXP(-ARG)
LGT = 1.0 + S-M*Y
RETURN
C 200 LGT = 1.0 + M-COS(2.0*PI*X - PH)
RETURN
END
SUBROUTINE BNDRY(T,X,U,UX,DBOU,DBOXX,DZDT,NPDE)
C
C DIMENSION U(NPDE),UX(NPDE),DZDT(NPDE)
C DIMENSION DBOU(NPDE),DBOXX(NPDE),DBOXX(NPDE)
C COMMON/A/DUM(3),G1,DVM(7)
C
C FROM THE BOUNDARY CONDITIONS OF THE FORM B(U,UX) = Z(T), FIND THE
C PARTIAL DERIVATIVES OF BOTH SIDES. THE BOUNDARY CONDITIONS ARE
C THE SAME AT XL AND XR.
C
C DBOU(1,1) = 1.0
C DBOU(1,2) = -1.0
C DBOU(2,1) = 0.0
C DBOU(2,2) = 0.0
C DBOXX(1,1) = 0.0
C DBOXX(1,2) = 0.0
C DBOXX(2,1) = 1.0
C DBOXX(2,2) = 0.0
C DZDT(1) = 0.0
C DZDT(2) = 0.0
C
C RETURN
C END
SUBROUTINE UINIT(X,U,NPDE)
C
C SET INITIAL CONDITIONS, REPRESENTING ZERO CONDUCTION-ELECTRON DENSITY
C ZERO OCCUPIED DONOR-LEVEL DENSITY
C
C DIMENSION U(NPDE)
C U(1) = 0.0
C U(2) = 0.0
C RETURN
C END
SUBROUTINE SCFLD(T,WORK)
C
C SUBROUTINE SCFLD EVALUATES THE ELECTRIC FIELD (REALLY, A NORMALIZED
C FIELD; SEE NOTES) DUE TO THE NET SPACE-CHARGE INSIDE THE CRYSTAL, BY
C INTEGRATING THE NET CHARGE DENSITY FROM XL (THE LEFT BOUNDARY) TO X.
C THE FIELD, ES(1), IS OBTAINED AT THE (NINT+1) BREAKPOINTS XBKPT(1),
C THE COMPLETE INTEGRAL IS DONE IN PIECES, CONSISTING OF THE INTERVALS
C BETWEEN CONSECUTIVE BREAKPOINTS. THE SUM OF ALL INTEGRALS UP TO AND
C INCLUDING THE 1TH ONE IS ES(1). THE BREAKPOINTS ARE ASSUMED TO BE E-
C QUALLY-SPACED. WHETHER THE FIELD IS STORED IN ES(1,2) OR ES(1,3) DE-
C PENDS ON WHETHER T > TOUT (K = 2) OR T < TOUT (K = 1). IN THE FORMER
C CASE, SCFLD IS CALLED FROM PDECOL; IN THE LATTER, FROM MAIN.
C
C DIMENSION XBKPT(201),E(201),ES(201,3),XPT(16),U(2,16,1),SCFCH(4)
C DIMENSION WORK(1),TE(3)
C DIMENSION CS(4,8),NS(4,8)
C COMMON/EFIELD/XBKPT,E,ES,TE,TPRV,TER,DIS,NPTS,INTR,ICNT

```

```

C      CCM/WM/MTS/CS.WS
C      (
      IRK = INTR/2
      INN = INTR/4
      SUM = 0.0
      K = 2
      IF (T.LT.TE(3)) K = 1
      DO 140 J = 1,K
        TE(J) = TE(J+1)
      DO 140 I = 1,NPTS
140   ES(I,J) = ES(I,J+1)
155   ES(I,K+1) = 0.0
      XU = XBKPT(1)
      B = XBKPT(2) - XBKPT(1)
C      LOOP TO COMPUTE THE INTEGRALS IN EACH OF THE INTERVALS
C
C      DO 200 I = 2,NPTS
C
C      SET INTEGRATION LIMITS FOR THE ITH INTERVAL
C
C      XL = XU
C      XU = XBKPT(I)
C
C      OBTAIN THE POINTS, XPT, AT WHICH THE ITH INTEGRAND WILL NEED TO
C      BE EVALUATED. INTR-POINT GAUSSIAN QUADRATURE WILL BE USED.
C
C      A = .5*(XU+XL)
C      DO 160 J = 1,IRR
C      C = CS(INN,J)*B
C      XPT(2*J-1) = A + C
160   XPT(2*J) = A - C
C
C      EVALUATE THE DEPENDENT VARIABLES, U (U AND V) AT THESE POINTS
C
C      CALL VALUES(XPT,U,SCCH,2,16,INTR,0,WORK)
C
C      EVALUATE THE INTEGRAND, V - U, AT EACH POINT REQUIRED BY THE
C      QUADRATURE FORMULA
C
C      DO 170 J = 1,INTR
170   U(2,J,1) = U(2,J,1) - U(1,J,1)
C
C      EVALUATE THE INTEGRAL BY SUMMING THE WEIGHTED FUNCTION VALUES
C
C      Y = 0.0
C      DO 180 J = 1,IRR
C      W = WS(INN,J)
180   Y = Y + W*(U(2,2*J-1,1) + U(2,2*J,1))
C      Y = Y*B
C
C      SUM, THE TOTAL INTEGRAL, INCLUDES CONTRIBUTIONS FROM INTERVALS 1,2,...I
C
C      SUM = SUM + Y
200   ES(1,K+1) = SUM
      IF (K.EQ.2) CALL MERR
      RETURN
      END
      SUBROUTINE EFIELD(X,T,Y)
C
C      SUBROUTINE TO CALCULATE THE SPACE-CHARGE FIELD AT X, T. ARRAY E,
C      WHICH IS SET UP IN ESTOR, CORRESPONDS TO THE (ANTICIPATED) FIELD AT
C      ALL THE BREAK-POINTS, AT THE TIME OF INTEREST. THE VALUE OF THE FIELD
C      AT X IS FOUND FROM THIS ARRAY BY INTERPOLATION.
C

```

```

C      DIMENSION E(201),XBKPT(201),ES(201,3),TE(3)
C      CC( N/EFIELD,XBKPT,E,ES,TE,TPRV,TER,DIS,NPTS,INTR,ICNT) (
C      CALL ESTOR TO UPDATE THE ARRAY. E, TO THE CURRENT TIME IF THIS HAS
C      NOT ALREADY BEEN DONE
C      ICNT = ICNT + 1
C      IF(T.EQ.TPRV)GO TO 90
C      CALL ESTOR(T)
C
C      SELECT THE RANGE (MIN, MAX) OF THE INDEX (I) TO BE SEARCHED FOR
C      THOSE TABLE VALUES (XBKPT(I)) NEAREST X.
C
C      90 NDIM = 3
C      EPS = 1.0E-03
C      Y = (X - XBKPT(1))/(XBKPT(NPTS) - XBKPT(1))
C      Y = Y*NPTS - 2
C      MIN = Y
C      IF(MIN.LT.1)MIN = 1
C      MAX = MIN + 6
C      IF(MAX.LE.NPTS)GO TO 100
C      MAX = NPTS
C      MIN = MAX - 6
C      100 CONTINUE
C
C      CALL THE SUBROUTINE TO DO THE NDIM-POINT INTERPOLATION. THE RESULTANT
C      VALUE IS Y. THE ARRAY WORK IN INT HAS NOTHING TO DO WITH THE ARRAY
C      OF THE SAME NAME IN THE OTHER ROUTINES.
C
C      CALL INT(E,XBKPT,X,MIN,MAX,NDIM,EPS,Y)
C      RETURN
C      END
C      SUBROUTINE ESTOR(T)
C
C      THIS SUBROUTINE TAKES THE CONTENTS OF ES (CONTAINING THE SPACE-CHARGE
C      FIELD AT THE XBKPT AT TIMES TE(1), TE(2), AND TE(3)) AND DOES AN INTER-
C      POLATION OR EXTRAPOLATION TO TIME T. THE RESULTS ARE STORED IN ARRAY E
C
C      DIMENSION E(201),ES(201,3),TE(3),EE(3),XBKPT(201)
C      COMMON/EFIELD/XBKPT,E,ES,TE,TPRV,TER,DIS,NPTS,INTR,ICNT
C      COMMON/GEARO/DTUSED,NQ,NSTEP,NF,NJ
C
C      EPS = 1.0E-03
C      IF(TPRV.LT.0.0)JCNT = 0
C      I = ICNT - JCNT
C      DTU = T - TPRV
C      TPRV = T
C      DT = T - TER
C      DO 90 J = 1,3
C      90 EE(J) = TE(J) - TER
C      WRITE(3,700)NSTEP,DT,DTU,ICNT,I,EE(1),EE(2),EE(3),DIS
C      700 FORMAT(1X,NSTEP=,I3.0,T=,E10.4,DT=,E10.4,ICNT=,I6,
C      115.2X,3E10.4,DIS=,E10.4)
C      JCNT = ICNT
C      DT = TE(2) - TE(1)
C      IF(DT.EQ.0.0)RETURN
C      DO 100 I = 1,NPTS
C      DO 95 J = 1,3
C      95 EE(J) = ES(I,J)
C      CALL INT(EE,TE,T,1,3,EPS,Y)
C      100 E(I) = Y
C      RETURN
C      END
C      SUBROUTINE RERR
C
C      DIMENSION XBKPT(201),E(201),ES(201,3),TE(3)

```

```

COMMON/EFIELD/XBKPT,E,ES,TE,TPRV,TER,DIS,NPTS,INTR,ICNT
C
C      CALCULATE THE AVERAGE ABSOLUTE ERROR IN THE SPACE-CHARGE FIELD USED IN
C      SUBROUTINE F AT THE CURRENT STEP BY COMPARING E (THE FIELD OBTAINED BY
C      EXTRAPOLATION FROM THE 'CORRECT' VALUES AT THE PREVIOUS TWO STEPS) WITH
C      ES (THE 'CORRECT' VALUE OBTAINED BY INTEGRATING RHO AFTER THE CURRENT
C      STEP IS COMPLETE.)
C
DIS = 0.0
DO 100 I = 1,NPTS
100 DIS = DIS + ABS(E(I) - ES(I,3))
DIS = DIS/NPTS
RETURN
END
SUBROUTINE INT(YS,XS,X,MIN,MAX,NDIM,EPS,Y)
C
C      SUBROUTINE TO SELECT AND ORDER THE NDIM POINTS TO BE USED FOR
C      AITKEN-LAGRANGE INTERPOLATION BY SUBROUTINE ALI
C
XS = NDIM ABSCISSAE NEAREST X
YS = NDIM ORDINATES CORRESPONDING TO XS
C
X = POINT AT WHICH FUNCTION IS TO BE EVALUATED BY INTERPOLATION
MIN,MAX = RANGE OF TABLE TO BE SEARCHED FOR NDIM POINTS NEAREST X
NDIM = NUMBER OF POINTS USED IN INTERPOLATION PROCEDURE
EPS = DESIRED ACCURACY OF INTERPOLATION
Y = RESULTANT FUNCTION VALUE
C
DIMENSION ARG(9),VAL(9)
DIMENSION YS(1),XS(1),WORK(201)
B = 0.0
DO 5 N = MIN,MAX
DEL = ABS(XS(N) - X)
IF(DEL.LT.B)GO TO 5
B = DEL
5 WORK(N) = DEL
B = B + 1.0
DO 10 J = 1,NDIM
DEL = B
DO 7 N = MIN,MAX
IF(WORK(N).GE.DEL)GO TO 7
1 = N
DEL = WORK(N)
7 CONTINUE
VAL(J) = YS(1)
ARG(J) = XS(1)
10 WORK(1) = B
CALL ALI(X,ARG,VAL,Y,EPS,NDIM,IER)
RETURN
END
SUBROUTINE ALI(X,ARG,VAL,Y,EPS,NDIM,IER)
C
C      SUBROUTINE ALI PERFORMS NDIM-POINT INTERPOLATION ACCORDING TO THE
C      AITKEN-LAGRANGE METHOD. IT IS DESCRIBED IN THE IBM SCIENTIFIC SUB-
C      ROUTINE PACKAGE.
C
X = POINT AT WHICH FUNCTION IS TO BE EVALUATED BY INTERPOLATION
ARG = NDIM ABSCISSAE NEAREST X
VAL = NDIM ORDINATES CORRESPONDING TO ARG
Y = DESIRED FUNCTION VALUE
EPS = DESIRED ACCURACY OF INTERPOLATION
NDIM = NUMBER OF POINTS USED IN INTERPOLATION PROCEDURE
IER = 0 IF SUBROUTINE OPERATES PROPERLY
C
DIMENSION ARG(9),VAL(9)

```



```

C      IF I = 2
C      DELT2 = 0.0
C      IF (NDIM - 1) 9,7,1
C
C      START OF AITKEN-LOOP
C
C      1 DO 6 J = 2,NDIM
C      DELT1 = DELT2
C      IEND = J - 1
C      DO 2 I = 1,IEND
C      H = ARG(I) - ARG(J)
C      IF (H) 2,13,2
C      2 VAL(J) = (VAL(I) - ARG(J)) - VAL(J) * (X - ARG(I)) / H
C      DELT2 = ABS(VAL(J) - VAL(IEND))
C      IF (J - 2) 6,6,3
C      3 IF (DELT2 - EPS) 10,10,4
C      4 IF (J - 6) 6,5,5
C      5 IF (DELT2 - DELT1) 6,11,11
C      6 CONTINUE
C
C      END OF AITKEN-LOOP
C
C      7 J = NDIM
C      8 Y = VAL(J)
C      9 RETURN
C
C      THERE IS SUFFICIENT ACCURACY WITHIN NDIM - 1 ITERATION STEPS
C
C      10 IER = 0
C      GO TO 8
C
C      TEST VALUE DELT2 STARTS OSCILLATING
C
C      11 IER = 1
C      12 J = IEND
C      GO TO 8
C
C      THERE ARE 2 IDENTICAL ARGUMENTS IN VECTOR ARG
C
C      13 IER = 3
C      GO TO 12
C
C      END
C      SUBROUTINE PEAK(Y,X,NO,YMAX,XMAX,EP,K)
C
C      DIMENSION Y(41),X(41)
C
C      FIND THE APPROXIMATE MAXIMUM OF A TABULATED FUNCTION, Y(X), BY INTER-
C      POLATION. THE POINTS, X, ARE ASSUMED EQUALLY SPACED. THE LOCATION OF
C      THE MAXIMUM, XMAX, IS FOUND TO WITHIN EP, AND YMAX IS THE APPROXIMATE
C      VALUE OF Y AT XMAX.
C
C      THE MINIMUM OF A TABULATED FUNCTION CAN BE FOUND BY SETTING K > 0. FOR
C      K = 0, THE MAXIMUM IS LOCATED.
C
C      IF (K.EQ.0) GO TO 145
C      DO 140 I = 1,NO
C      140 Y(I) = -Y(I)
C      145 NDIM = 3
C      EPS = 1.0E-03
C      NINT = 20
C      YMAX = -1.0E+321
C      DO 150 I = 1,NO
C      Q = Y(I)
C      IF (Q.LE.YMAX) GO TO 150
C      YMAX = Q

```

```

150 19 I
    .NUE
    IF(IC.EQ.NO)IC = NO - 1
    IF(IC.EQ.1)IC = 2
    DX = (X(2)-X(1))/NINT
    DO 160 I = 1,5
    IF(DX.LE.EP)GO TO 170
    NINT = NINT*2
160 DX = DX/2.0
170 XMAX = X(IC)
    Q = (Y(IC-1) - Y(IC+1))/Y(IC)
    MIN = IC - 1
    MAX = IC + 1
    IF(ABS(Q).LT.EPS)GO TO 210
    IF(Q.GT.0.0)IC = IC - 1
    Q = Y(IC)
    DO 180 I = 1,NINT
    XMAX = X(IC) + I*DX
    CALL INT(Y,X,XMAX,MIN,MAX,NDIM,EPS,YMAX)
    IF(YMAX-Q)190,200,180
180 Q = YMAX
    DX = 0.0
190 XMAX = XMAX - DX/2.0
200 XMAX = XMAX - DX/2.0
210 CALL INT(Y,X,XMAX,MIN,MAX,NDIM,EPS,YMAX)

    IF(K.EQ.0)RETURN
    DO 220 I = 1,NO
220 Y(I) = -Y(I)
    YMAX = -YMAX
    RETURN
    END

```

C

```

C
C      (
C      SUBROUTINE PDECOL (TO, TOUT, DT, XBKPT, EPS, NINT, KORD, NCC, NPDE, MF,
C      INDEX, WORK, IWORK)
C      -----
C
C      THIS IS THE MARCH 24, 1978 VERSION OF PDECOL.
C
C      THIS PACKAGE WAS CONSTRUCTED SO AS TO CONFORM TO AS MANY ANSI-FORTRAN
C      RULES AS WAS CONVENIENTLY POSSIBLE. THE FORTRAN USED VIOLATES ANSI
C      STANDARDS IN THE TWO WAYS LISTED BELOW....
C
C      1. SUBSCRIPTS OF THE GENERAL FORM C*V1 + V2 + V3 ARE USED
C      (POSSIBLY IN A PERMUTED ORDER), WHERE C IS AN INTEGER CONSTANT
C      AND V1, V2, AND V3 ARE INTEGER VARIABLES.
C
C      2. ARRAY NAMES APPEAR SINGLY IN DATA STATEMENTS IN THE ROUTINES
C      BSPLVN AND COSET.
C
C      MACHINE DEPENDENT FEATURES.....
C
C      THIS VERSION OF PDECOL WAS DESIGNED FOR USE ON CDC MACHINES WITH
C      A WORD LENGTH OF 60 BITS. WE DO NOT RECOMMEND THE USE OF PDECOL WITH
C      WORD LENGTHS OF LESS THAN 48 BITS. THE MOST IMPORTANT MACHINE
C      AND WORD LENGTH DEPENDENT CONSTANTS ARE DEFINED IN THE BLOCK DATA
C      AND IN SUBROUTINES COLPNT AND COSET. THE USER SHOULD CHECK THESE
C      CAREFULLY FOR APPROPRIATENESS FOR HIS LOCAL SITUATION. THE FORTRAN
C      FUNCTIONS USED BY EACH ROUTINE ARE LISTED IN THE COMMENTS TO
C      FACILITATE CONVERSION TO DOUBLE PRECISION.
C
C      -----
C
C      PDECOL IS THE DRIVER ROUTINE FOR A SOPHISTICATED PACKAGE OF
C      SUBROUTINES WHICH IS DESIGNED TO SOLVE THE GENERAL SYSTEM OF
C      NPDE NONLINEAR PARTIAL DIFFERENTIAL EQUATIONS OF AT MOST SECOND
C      ORDER ON THE INTERVAL (XLEFT,XRIGHT) FOR T .GT. TO WHICH IS OF THE
C      FORM....
C
C      DU/DT = F ( T, X, U, UX, UXX )
C
C      WHERE
C
C      U = ( U(1), U(2), ... , U(NPDE) )
C      UX = ( UX(1), UX(2), ... , UX(NPDE) )
C      UXX = ( UXX(1), UXX(2), ... , UXX(NPDE) )
C
C      EACH U(K) IS A FUNCTION OF THE SCALAR QUANTITIES T AND X.
C      U(K) REPRESENTS THE FIRST PARTIAL DERIVATIVE OF U(K) WITH RESPECT
C      TO THE VARIABLE X. UXX(K) REPRESENTS THE SECOND PARTIAL DERIVATIVE
C      OF U(K) WITH RESPECT TO THE VARIABLE X, AND DU/DT IS THE VECTOR OF
C      PARTIAL DERIVATIVES OF U WITH RESPECT TO THE TIME VARIABLE T.
C      F REPRESENTS AN ARBITRARY VECTOR VALUED FUNCTION WHOSE NEDS
C      COMMENTS DEFINE THE RESPECTIVE PARTIAL DIFFERENTIAL EQUATIONS OF
C      THE PDE SYSTEM. SEE SUBROUTINE F DESCRIPTION BELOW.
C
C      BOUNDARY CONDITIONS
C
C      DEPENDS ON THE TYPE OF PDE(S), 0, 1, OR 2 BOUNDARY CONDITIONS
C      ARE REQUIRED FOR EACH PDE IN THE SYSTEM. THESE ARE IMPOSED AT XLEFT
C      AND/OR XRIGHT AND EACH MUST BE OF THE FORM....
C

```



```

C      CONDITION. THIS ROUTINE CAN BE STRUCTURED AS FOLLOWS...
C      ( IF NOT BE SUPPLIED AND DEFINED BY THE USER. )
C      COMMON ENDST = LEFT
C      IF ( X .NE. XLEFT ) GO TO 10
C      HERE DEFINE AND SET PROPER VALUES FOR DBOUX(K,J), OBOUX(K,J),
C      AND ODOUX(K,J) FOR K,J = 1 TO NPDE FOR THE LEFT BOUNDARY POINT
C      X = XLEFT.
C      RETURN
C 10 CONTINUE
C      HERE DEFINE AND SET PROPER VALUES FOR DBOUX(K,J), OBOUX(K,J),
C      AND ODOUX(K,J) FOR K,J = 1 TO NPDE FOR THE RIGHT BOUNDARY POINT
C      X = XRIGHT.
C      RETURN
C      END
C 3) SUBROUTINE UINIT( X, U, NPDE )
C      DIMENSION U(NPDE)
C      THIS ROUTINE IS USED TO PROVIDE THE PDE PACKAGE WITH THE
C      NEEDED INITIAL CONDITION FUNCTION VALUES. THE PACKAGE
C      PROVIDES A VALUE OF THE INPUT VARIABLE X, AND THE USER IS TO
C      DEFINE THE PROPER INITIAL VALUES (AT T = T0) FOR ALL OF THE
C      PDE COMPONENTS, I.E.
C      U(X) = DESIRED INITIAL VALUE OF PDE COMPONENT U(X) AT
C      X AND T = T0 FOR K = 1 TO NPDE.
C      NOTE... THE INCOMING VALUE OF X WILL BE A COLLOCATION POINT
C      VALUE. THE INITIAL CONDITIONS AND BOUNDARY CONDITIONS
C      MUST BE CONSISTENT (SEE ABOVE).
C      RETURN
C      END

```

231

OPTIONAL USER SUPPLIED SUBROUTINE

```

C IF THE USER DESIRES TO USE THE MF = 11 OR 21 OPTION IN ORDER TO SAVE
C ABOUT 10-20 PERCENT IN EXECUTION TIME (SEE BELOW), THEN THE USER MUST
C PROVIDE THE FOLLOWING SUBROUTINE WHICH PROVIDES INFORMATION ABOUT THE
C DERIVATIVES OF THE FUNCTION F ABOVE. THIS PROVIDES FOR MORE EFFICIENT
C JACOBIAN MATRIX GENERATION. ON MOST COMPUTER SYSTEMS, THE USER WILL
C BE REQUIRED TO SUPPLY THIS SUBROUTINE AS A DUMMY SUBROUTINE IF THE
C OPTIONS MF = 12 OR 22 ARE USED (SEE BELOW).
C 1) SUBROUTINE DERIVE( T, X, U, UX, DFDU, DFDUX, DFDUXX, NPDE )
C      DIMENSION U(NPDE), UX(NPDE), DFDU(NPDE), DFDUX(NPDE), DFDUXX(NPDE)
C      THE PACKAGE PROVIDES VALUES OF THE INPUT VARIABLES T, X, U, UX,
C      AND DFDU. AND THE USER SHOULD CONSTRUCT THIS ROUTINE TO PROVIDE
C      THE FOLLOWING CORRESPONDING VALUES OF THE OUTPUT ARRAYS
C      DFDU, DFDUX, AND DFDUXX FOR K,J = 1 TO NPDE...
C      DFDU(K,J) = PARTIAL DERIVATIVE OF THE K-TH COMPONENT OF THE
C      PDE DEFINING FUNCTION F WITH RESPECT TO THE
C      VARIABLE U(J).
C      DFDUX(K,J) = PARTIAL DERIVATIVE OF THE K-TH COMPONENT OF THE
C      PDE DEFINING FUNCTION F WITH RESPECT TO THE
C      VARIABLE UX(J).
C      DFDUXX(K,J) = PARTIAL DERIVATIVE OF THE K-TH COMPONENT OF THE
C      PDE DEFINING FUNCTION F WITH RESPECT TO THE
C      VARIABLE UX(J).
C      NOTE... THE INCOMING VALUE OF X WILL BE A COLLOCATION POINT
C      VALUE.
C      RETURN
C      END

```

METHODS USED

THE PACKAGE PDECOL IS BASED ON THE METHOD OF LINES AND USES A
 FINITE ELEMENT COLLOCATION PROCEDURE WITH PIECEWISE POLYNOMIAL
 AS TRIAL SPACE FOR THE DISCRETIZATION OF THE SPATIAL VARIABLE
 X. THE COLLOCATION PROCEDURE REDUCES THE PDE SYSTEM TO A SEMI-
 DISCRETE SYSTEM WHICH THEN DEPENDS ONLY ON THE TIME VARIABLE T.
 THE TIME INTEGRATION IS THEN ACCOMPLISHED BY USE OF SLIGHTLY
 MODIFIED STANDARD TECHNIQUES (SEE REFS. 1,2).

PIECEWISE POLYNOMIALS

THE USER IS REQUIRED TO SELECT THE PIECEWISE POLYNOMIAL SPACE
 WHICH IS TO BE USED TO COMPUTE HIS APPROXIMATE SOLUTION. FIRST, THE
 ORDER, KORD, OF THE POLYNOMIALS TO BE USED MUST BE SPECIFIED
 (KORD = POLYNOMIAL DEGREE + 1). NEXT, THE NUMBER OF PIECES
 (INTERVALS) NINT, INTO WHICH THE SPATIAL DOMAIN (XLEFT,XRIGHT) IS
 TO BE DIVIDED, IS CHOSEN. THE NINT + 1 DISTINCT BREAKPOINTS OF
 THE DOMAIN MUST BE DEFINED AND SET INTO THE ARRAY XBKPT IN
 STRICTLY INCREASING ORDER, I.E.
 C XLEFT,XBKPT(1) .LT. XBKPT(2) .LT. LT. XBKPT(NINT+1)=XRIGHT.
 C THE APPROXIMATE SOLUTION AT ANY TIME T WILL BE A POLYNOMIAL OF
 C ORDER KORD OVER EACH SUBINTERVAL (XBKPT(I),XBKPT(I+1)). THE
 C NUMBER OF CONTINUITY CONDITIONS, NCC, TO BE IMPOSED ACROSS ALL OF
 C THE BREAKPOINTS IS THE LAST PIECE OF USER SUPPLIED DATA WHICH IS
 C REQUIRED TO UNIQUELY DETERMINE THE DESIRED PIECEWISE POLYNOMIAL
 C SPACE. FOR EXAMPLE, NCC = 2 WOULD REQUIRE THAT THE APPROXIMATE
 C SOLUTION (MADE UP OF THE SEPARATE POLYNOMIAL PIECES) AND ITS FIRST
 C SPATIAL DERIVATIVE BE CONTINUOUS AT THE BREAKPOINTS AND HENCE ON
 C THE ENTIRE DOMAIN (XLEFT,XRIGHT). NCC = 3 WOULD REQUIRE THAT THE
 C APPROXIMATE SOLUTION AND ITS FIRST AND SECOND SPATIAL DERIVATIVES
 C BE CONTINUOUS AT THE BREAKPOINTS, ETC. THE DIMENSION OF THIS LINEAR
 C SPACE IS KNOWN AND FINITE AND IS NCPTS = KORD*NINT - NCC*(NINT-1).
 C THE WELL-KNOWN B-SPLINE BASIS (SEE REF. 3) FOR THIS SPACE IS USED
 C BY PDECOL AND IT CONSISTS OF NCPTS KNOWN PIECEWISE POLYNOMIAL
 C FUNCTIONS BF(I,X) FOR I=1 TO NCPTS, WHICH DO NOT DEPEND ON THE
 C TIME VARIABLE T. WE WISH TO EMPHASIZE THAT THE PIECEWISE POLYNOMIAL
 C SPACE USED IN PDECOL (WHICH IS SELECTED BY THE USER) WILL DETERMINE
 C THE MAGNITUDE OF THE SPATIAL DISCRETIZATION ERRORS IN THE COMPUTED
 C APPROXIMATE SOLUTION. THE PACKAGE HAS NO CONTROL OVER ERRORS
 C INTRODUCED BY THE USER'S CHOICE OF THIS SPACE. SEE INPUT PARAMETERS
 C BELOW.

COLLOCATION OVER PIECEWISE POLYNOMIALS

THE BASIC ASSUMPTION MADE IS THAT THE APPROXIMATE SOLUTION
 SATISFIES

$$U(T,X) = \sum_{i=1}^{NCPTS} C(I,T) * BF(I,X)$$

WHERE THE UNKNOWN COEFFICIENTS C DEPEND ONLY ON THE TIME T AND
 THE KNOWN BASIS FUNCTIONS DEPEND ONLY ON X (WE HAVE ASSUMED THAT
 NCPTS = 1 FOR CONVENIENCE). SO, AT ANY GIVEN TIME T THE APPROX-
 IMATE SOLUTION IS A PIECEWISE POLYNOMIAL IN THE USER CHOSEN SPACE.
 THE SEMI-DISCRETE EQUATIONS (ACTUALLY ORDINARY DIFFERENTIAL
 EQUATIONS) WHICH DETERMINE THE COEFFICIENTS C ARE OBTAINED BY
 REQUIRING THAT THE ABOVE APPROXIMATE U(T,X) SATISFY THE PDE AND
 BOUNDARY CONDITIONS EXACTLY AT A SET OF NCPTS COLLOCATION POINTS
 (SEE COLPNT). THUS, PDECOL ACTUALLY COMPUTES THE BASIS FUNCTION
 COEFFICIENTS RATHER THAN SPECIFIC APPROXIMATE SOLUTION VALUES.

REFERENCES

1. MADSEN, N.K. AND R.F. SINCOVEC. PDECOL - COLLOCATION SOFTWARE
 FOR PARTIAL DIFFERENTIAL EQUATIONS. ACM-TOMS, VOL. , NO. ,
2. SINCOVEC, R.F. AND N.K. MADSEN, SOFTWARE FOR NONLINEAR PARTIAL

DIFFERENTIAL EQUATIONS, ACM-TOMS, VOL. 1, NO. 3,
 SEPTEMBER 1975, PP. 232-260.
 3. H. ADAMS, A.C., PRELIMINARY DOCUMENTATION OF GEARIB, SOLUTION
 OF IMPLICIT SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS WITH
 BAYDED JACOBIANS, LAWRENCE LIVERMORE LAB. UCID-30130, FEBRUARY
 1976.
 4. DEBOOR, C., PACKAGE FOR CALCULATING WITH B-SPLINES, SIAM J.
 NUMER. ANAL., VOL. 14, NO. 3, JUNE 1977, PP. 441-472.

USE OF POECOL

POECOL IS CALLED ONCE FOR EACH DESIRED OUTPUT VALUE (TOUT) OF THE
 TIME T, AND IT IN TURN MAKES REPEATED CALLS TO THE CORE INTEGRATOR,
 STIFB, WHICH ADVANCES THE TIME BY TAKING SINGLE STEPS UNTIL
 T.GE. TOUT. INTERPOLATION TO THE EXACT TIME TOUT IS THEN DONE.
 SEE TOUT BELOW.

SUMMARY OF SUGGESTED INPUT VALUES

IT IS OF COURSE IMPOSSIBLE TO SUGGEST INPUT PARAMETER VALUES WHICH
 ARE APPROPRIATE FOR ALL PROBLEMS. THE FOLLOWING SUGGESTIONS ARE TO
 BE USED ONLY IF YOU HAVE NO IDEA OF BETTER VALUES FOR YOUR PROBLEM.

DT = 1.E-10
 XBKPT = CHOOSE NINT+1 EQUALLY SPACED VALUES SUCH THAT XBKPT(1) =
 XLEFT AND XBKPT(NINT+1) = XRIGHT.
 EPS = 1.E-4
 NINT = ENOUGH SO THAT ANY FINE STRUCTURE OF THE PROBLEM MAY BE
 RESOLVED.
 KORO = 4
 NCC = 2
 MP = 22
 INDEX = 1 (ON FIRST CALL ONLY, THEN 0 THEREAFTER).

THE INPUT PARAMETERS ARE..

TO = THE INITIAL VALUE OF T, THE INDEPENDENT VARIABLE
 (USED ONLY ON FIRST CALL).
 TOUT = THE VALUE OF T AT WHICH OUTPUT IS DESIRED NEXT. SINCE
 THE PACKAGE CHOOSES ITS OWN TIME STEP SIZES, THE
 INTEGRATION WILL NORMALLY GO SLIGHTLY BEYOND TOUT
 AND THE PACKAGE WILL INTERPOLATE TO T = TOUT.
 DT = THE INITIAL STEP SIZE IN T, IF INDEX = 1, OR, THE
 MAXIMUM STEP SIZE ALLOWED (MUST BE .GT. 0). IF INDEX = 3,
 USED FOR INPUT ONLY WHEN INDEX = 1 OR 3. SEE BELOW.
 XBKPT = THE ARRAY OF PIECEWISE POLYNOMIAL BREAKPOINTS.
 THE NINT+1 VALUES MUST BE STRICTLY INCREASING WITH
 XBKPT(1) = XLEFT AND XBKPT(NINT+1) = XRIGHT (USED ONLY
 ON FIRST CALL).
 EPS = THE RELATIVE TIME ERROR BOUND (USED ONLY ON THE
 FIRST CALL, UNLESS INDEX = 4). SINGLE STEP ERROR
 ESTIMATES DIVIDED BY CMAX(I) WILL BE KEPT LESS THAN
 EPS IN ROOT-MEAN-SQUARE NORM. THE VECTOR CMAX OF WEIGHTS
 IS COMPUTED IN POECOL. INITIALLY CMAX(I) IS SET TO
 ABS(C(I)), WITH A DEFAULT VALUE OF 1 IF ABS(C(I)) .LT. 1.
 THEREAFTER, CMAX(I) IS THE LARGEST VALUE
 OF ABS(C(I)) SEEN SO FAR, OR THE INITIAL CMAX(I) IF
 THAT IS LARGER. TO ALTER EITHER OF THESE, CHANGE THE
 APPROPRIATE STATEMENTS IN THE DO-LOOPS ENDING AT
 STATEMENTS 50 AND 130 BELOW. THE USER SHOULD EXERCISE
 SOME DISCRETION IN CHOOSING EPS. IN GENERAL, THE
 OVERALL RUNNING TIME FOR A PROBLEM WILL BE GREATER IF
 EPS IS CHOSEN SMALLER. THERE IS USUALLY LITTLE REASON TO
 CHOOSE EPS MUCH SMALLER THAN THE ERRORS WHICH ARE BEING

INTRODUCED BY THE USER'S CHOICE OF THE POLYNOMIAL SPACE.
 SEE RELATED COMMENTS CONCERNING CMAX BELOW STATEMENT 40.
 NINT = THE NUMBER OF SUBINTERVALS INTO WHICH THE SPATIAL DL IN
 (XLEFT, XRIGHT) IS TO BE DIVIDED (MUST BE .GE. 1)
 (USED ONLY ON FIRST CALL).
 KORD = THE ORDER OF THE PIECEWISE POLYNOMIAL SPACE TO BE USED.
 ITS VALUE MUST BE GREATER THAN 2 AND LESS THAN 21. FOR
 FIRST ATTEMPTS WE RECOMMEND KORD = 4. IF THE SOLUTION
 IS SMOOTH AND MUCH ACCURACY IS DESIRED, HIGHER VALUES
 MAY PROVE TO BE MORE EFFICIENT. WE HAVE SELDOM USED
 VALUES OF KORD IN EXCESS OF 8 OR 9, THOUGH THEY ARE
 AVAILABLE FOR USE IN PDECOL (USED ONLY ON FIRST CALL).
 NCC = THE NUMBER OF CONTINUITY CONDITIONS TO BE IMPOSED ON THE
 APPROXIMATE SOLUTION AT THE BREAKPOINTS IN AKRPT.
 NCC MUST BE GREATER THAN 1 AND LESS THAN KORD. WE
 RECOMMEND THE USE OF NCC = 2.
 NPDE = SINCE THEORY PREDICTS THAT DRAMATICALLY MORE
 ACCURATE RESULTS CAN OFTEN BE OBTAINED USING THIS CHOICE
 (USED ONLY ON FIRST CALL).
 MF = THE NUMBER OF PARTIAL DIFFERENTIAL EQUATIONS IN THE SYSTEM
 TO BE SOLVED (USED ONLY ON FIRST CALL).
 INDEX = THE METHOD FLAG (USED ONLY ON FIRST CALL, UNLESS
 INDEX = -1). ALLOWED VALUES ARE 11, 12, 21, 22.
 FOR FIRST ATTEMPTS WE RECOMMEND THE USE OF MF = 22.
 (MF HAS TWO DECIMAL DIGITS, METH AND MITER
 (MF = 10*METH + MITER).
 METH IS THE BASIC METHOD INDICATOR..
 METH = 1 MEANS THE ADAMS METHODS (GENERALIZATIONS OF
 CRANK-NICOLSON).
 METH = 2 MEANS THE BACKWARD DIFFERENTIATION
 FORMULAS (BDF), OR STIFF METHODS OF GEAR.
 MITER IS THE ITERATION METHOD INDICATOR
 AND DETERMINES HOW THE JACOBIAN MATRIX IS
 TO BE COMPUTED..
 MITER = 1 MEANS CHORD METHOD WITH ANALYTIC JACOBIAN.
 FOR THIS USER SUPPLIES SUBROUTINE DERIVF.
 SEE DESCRIPTION ABOVE.
 MITER = 2 MEANS CHORD METHOD WITH JACOBIAN CALCULATED
 INTERNALLY BY FINITE DIFFERENCES. SEE
 SUBROUTINES PSETIB AND DIFF.
 INDEX = INTEGER USED ON INPUT TO INDICATE TYPE OF CALL,
 WITH THE FOLLOWING VALUES AND MEANINGS..
 0 THIS IS THE FIRST CALL FOR THIS PROBLEM.
 1 THIS IS NOT THE FIRST CALL FOR THIS PROBLEM,
 AND INTEGRATION IS TO CONTINUE.
 2 SAME AS 0 EXCEPT THAT TOUT IS TO BE HIT
 EXACTLY (NO INTERPOLATION IS DONE). SEE NOTE
 BELOW. ASSUMES TOUT .GE. THE CURRENT T.
 IF TOUT IS .LT. THE CURRENT TIME, THEN TOUT IS
 RESET TO THE CURRENT TIME AND CONTROL IS
 RETURNED TO THE USER. A CALL TO VALUES WILL
 PRODUCE ANSWERS FOR THE NEW VALUE OF TOUT.
 3 SAME AS 0 EXCEPT CONTROL RETURNS TO CALLING
 PROGRAM AFTER ONE STEP. TOUT IS IGNORED AND
 DT MUST BE SET .GT. 0 TO A MAXIMUM ALLOWED
 DT VALUE.. SEE ABOVE.
 4 THIS IS NOT THE FIRST CALL FOR THE PROBLEM,
 AND THE USER HAS RESET EPS AND/OR MF.
 SINCE THE NORMAL OUTPUT VALUE OF INDEX IS 0,
 IT NEED NOT BE RESET FOR NORMAL CONTINUATION.
 NOTE.. THE PACKAGE MUST HAVE TAKEN AT LEAST ONE SUCCESSFUL TIME
 STEP BEFORE A CALL WITH INDEX = 2 OR 4 IS ALLOWED.
 AFTER THE INITIAL CALL, IF A NORMAL RETURN OCCURRED AND A NORMAL
 CONTINUATION IS DESIRED, SIMPLY RESET TOUT AND CALL AGAIN.
 ALL OTHER PARAMETERS WILL BE READY FOR THE NEXT CALL.


```

C A CHANGE OF PARAMETERS WITH INDEX = 4 CAN BE MADE AFTER
C EITHER SUCCESSFUL OR AN UNSUCCESSFUL RETURN PROVIDED AT LEAST
C ONE SUCCESSFUL TIME STEP HAS BEEN MADE.
C
C WORK = FLOATING POINT WORKING ARRAY FOR PDECOL. WE RECOMMEND
C THAT IT BE INITIALIZED TO ZERO BEFORE THE FIRST CALL
C TO PDECOL. ITS TOTAL LENGTH MUST BE AT LEAST
C
C KORD = 4*NPDE + 9*NPDE**2 + NCPTS*(3*KORD + 2) +
C NPDE*NCPTS*(3*ML + MAXDER + 7)
C
C WHERE ML AND MAXDER ARE DEFINED BELOW (SEE STORAGE
C ALLOCATION).
C
C IWORK = INTEGER WORKING ARRAY FOR PDECOL. THE FIRST TWO
C LOCATIONS MUST BE DEFINED AS FOLLOWS...
C IWORK(1) = LENGTH OF USERS ARRAY IWORK
C IWORK(2) = LENGTH OF USERS ARRAY IWORK
C THE TOTAL LENGTH OF IWORK MUST BE AT LEAST
C NCPTS*(NPDE + 1).
C
C OUTPUT
C
C THE SOLUTION VALUES ARE NOT RETURNED DIRECTLY TO THE USER BY PDECOL.
C THE METHODS USED IN PDECOL COMPUTE BASIS FUNCTION COEFFICIENTS. SO
C THE USER (AFTER A RETURN FROM PDECOL) MUST CALL THE PACKAGE ROUTINE
C VALUES TO OBTAIN HIS APPROXIMATE SOLUTION VALUES AT ANY DESIRED SPACE
C POINTS X AT THE TIME T = TOUT. SEE THE COMMENTS IN SUBROUTINE VALUES
C FOR DETAILS ON HOW TO PROPERLY MAKE THE CALL.
C
C EXECUTION ERROR MESSAGES WILL BE PRINTED BY PDECOL ON LOGICAL UNIT
C LOUT WHICH IS THE ONLY VARIABLE IN THE COMMON BLOCK /LOUNIT/. A
C DEFAULT OF LOUT = 3 IS SET IN THE BLOCK DATA.
C
C THE COMMON BLOCK /GEAR/ CONTAINS THE VARIABLES DTUSED, NUSED,
C NSTEP, NFE, AND NLE AND CAN BE ACCESSED EXTERNALLY BY THE USER IF
C DESIRED. RESPECTIVELY, IT CONTAINS THE STEP SIZE LAST USED (SUCCESS-
C FULLY), THE ORDER LAST USED (SUCCESSFULLY), THE NUMBER OF STEPS TAKEN
C SO FAR, THE NUMBER OF RESIDUAL EVALUATIONS (RES CALLS) SO FAR,
C AND THE NUMBER OF MATRIX EVALUATIONS (PSETIB CALLS) SO FAR.
C DIFFUN CALLS ARE COUNTED IN WITH RESIDUAL EVALUATIONS.
C
C THE OUTPUT PARAMETERS ARE..
C DT = THE STEP SIZE USED LAST, WHETHER SUCCESSFULLY OR NOT.
C TOUT = THE OUTPUT VALUE OF T. IF INTEGRATION WAS SUCCESSFUL,
C AND THE INPUT VALUE OF INDEX WAS NOT 3, TOUT IS
C UNCHANGED FROM ITS INPUT VALUE. OTHERWISE, TOUT
C IS THE CURRENT VALUE OF T TO WHICH THE INTEGRATION
C HAS BEEN COMPLETED.
C INDEX = INTEGER USED ON OUTPUT TO INDICATE RESULTS.
C WITH THE FOLLOWING VALUES AND MEANINGS...
C 0 INTEGRATION WAS COMPLETED TO TOUT OR BEYOND.
C -1 THE INTEGRATION WAS HALTED AFTER FAILING TO PASS THE
C ERROR TEST EVEN AFTER REDUCING DT BY A FACTOR OF
C 1.E10 FROM ITS INITIAL VALUE.
C -2 AFTER SOME INITIAL SUCCESS, THE INTEGRATION WAS
C HALTED EITHER BY REPEATED ERROR TEST FAILURES OR BY
C A TEST ON EPS. TOO MUCH ACCURACY HAS BEEN REQUESTED.
C -3 THE INTEGRATION WAS HALTED AFTER FAILING TO ACHIEVE
C CORRECTOR CONVERGENCE EVEN AFTER REDUCING DT BY A
C FACTOR OF 1.E10 FROM ITS INITIAL VALUE.
C -4 SINGULAR MATRIX ENCOUNTERED. PROBABLY DUE TO STORAGE
C OVERWRITES.
C -5 INDEX WAS 4 ON INPUT, BUT THE DESIRED CHANGES OF
C PARAMETERS WERE NOT IMPLEMENTED BECAUSE TOUT
C WAS NOT BEYOND T. INTERPOLATION TO T = TOUT WAS
C PERFORMED AS ON A NORMAL RETURN. TO TRY AGAIN,
C SIMPLY CALL AGAIN WITH INDEX = 4 AND A NEW TOUT.

```

```

C      ILLEGAL INDEX VALUE.
C      ILLEGAL EPS VALUE.
C      -8  AN ATTEMPT TO INTEGRATE IN THE WRONG DIRECTION. THE
C          SIGN OF DT IS WRONG RELATIVE TO TO AND TOUT.
C      -9  DT .EQ. 0.0.
C      -10 ILLEGAL NINT VALUE.
C      -11 ILLEGAL KORD VALUE.
C      -12 ILLEGAL NCC VALUE.
C      -13 ILLEGAL NPDE VALUE.
C      -14 ILLEGAL MF VALUE.
C      -15 ILLEGAL BREAKPOINTS - NOT STRICTLY INCREASING.
C      -16 INSUFFICIENT STORAGE FOR WORK OR IWORK.
C-----
C
C SUMMARY OF ALL PACKAGE ROUTINES
C
C PDECL - STORAGE ALLOCATION, ERROR CHECKING, INITIALIZATION, REPEATED
C         CALLS TO STIFB TO ADVANCE THE TIME.
C
C INTERP - INTERPOLATES COMPUTED BASIS FUNCTION COEFFICIENTS TO THE
C          DESIRED OUTPUT TIMES, TOUT, FOR USE BY VALUES.
C
C INITAL - INITIALIZATION, GENERATION AND STORAGE OF PIECEWISE POLY-
C          NOMIAL SPACE BASIS FUNCTION VALUES AND DERIVATIVES. DETER-
C          MINES THE BASIS FUNCTION COEFFICIENTS OF THE PIECEWISE
C          POLYNOMIALS WHICH INTERPOLATE THE USERS INITIAL CONDITIONS.
C
C COLPNT - GENERATION OF REQUIRED COLLOCATION POINTS.
C
C BSPLVD - B-SPLINE PACKAGE ROUTINES WHICH ALLOW FOR EVALUATION OF
C          ANY B-SPLINE BASIS FUNCTION OR DERIVATIVE VALUE.
C
C BSPLVN -
C          INTERV
C
C VALUES - GENERATION AT ANY POINT(S) OF VALUES OF THE COMPUTED
C           APPROXIMATE SOLUTION AND ITS DERIVATIVES WHICH ARE
C           PIECEWISE POLYNOMIALS. THE SUBROUTINE IS CALLED ONLY BY
C           THE USER.
C
C STIFB - CORE INTEGRATOR. TAKES SINGLE TIME STEPS TO ADVANCE THE
C         TIME. ASSEMBLES AND SOLVES THE PROPER NONLINEAR EQUATIONS
C         WHICH ARE RELATED TO USE OF ADAMS OR GEAR TYPE INTEGRATION
C         FORMULAS. CHOOSES PROPER STEP SIZE AND INTEGRATION FORMULA
C         ORDER TO MAINTAIN A DESIRED ACCURACY. DESIGNED FOR ODE
C         PROBLEMS OF THE FORM  $A * (DY/DT) = G(T,Y)$ .
C
C COSET - GENERATES INTEGRATION FORMULA AND ERROR CONTROL COEFFICIENTS.
C
C RES - COMPUTES RESIDUAL VECTORS USED IN SOLVING THE NONLINEAR
C        EQUATIONS BY A MODIFIED NEWTON METHOD.
C
C DIFFUN - COMPUTES  $A_{i+1} * G(T,Y)$  WHERE A AND G ARE AS ABOVE (STIFB).
C
C ADDA - ADDS THE A MATRIX TO A GIVEN MATRIX IN BAND FORM.
C
C EVAL - EVALUATES THE COMPUTED PIECEWISE POLYNOMIAL SOLUTION AND
C         DERIVATIVES AT COLLOCATION POINTS.
C
C GFUN - EVALUATES THE FUNCTION  $G(T,Y)$  BY CALLING EVAL AND THE USER
C         SUBROUTINES F AND BNDRY.
C
C PSETIB - GENERATES PROPER JACOBIAN MATRICES REQUIRED BY THE MODIFIED
C          NEWTON METHOD.
C
C DIFFF - PERFORMS SAME ROLE AS THE USER ROUTINE DERIVF. COMPUTES
C          DERIVATIVE APPROXIMATIONS BY USE OF FINITE DIFFERENCES.

```

```

C DECB ( PERFORM AN LU DECOMPOSITION AND FORWARD AND BACKWARD (
C SOLB SUBSTITUTION FOR SOLVING BANDED SYSTEMS OF LINEAR EQUATIONS.
C -----
C
C STORAGE ALLOCATION
C
C SINCE PDECOL IS A DYNAMICALLY DIMENSIONED PROGRAM, MOST OF ITS
C WORKING STORAGE IS PROVIDED BY THE USER IN THE ARRAYS WORK AND IWORK.
C THE FOLLOWING GIVES A LIST OF THE ARRAYS WHICH MAKE UP THE CONTENTS
C OF WORK AND IWORK, THEIR LENGTHS, AND THEIR USES. WHEN MORE THAN ONE
C NAME IS GIVEN, IT INDICATES THAT DIFFERENT NAMES ARE USED FOR THE
C SAME ARRAY IN DIFFERENT PARTS OF THE PROGRAM. THE DIFFERENT NAMES
C OCCUR BECAUSE PDECOL IS AN AMALGAMATION OF SEVERAL OTHER CODES
C WRITTEN BY DIFFERENT PEOPLE AND WE HAVE TRIED TO LEAVE THE SEPARATE
C PARTS AS UNCHANGED FROM THEIR ORIGINAL VERSIONS AS POSSIBLE.
C
C NAMES LENGTH USE
C -----
C
C BC 4*NPDE**2 BOUNDARY CONDITION INFORMATION.
C WORK
C
C A 3*WORD*NCPTS BASIS FUNCTION VALUES AT COLLOCATION POINT
C WORK(IW1)
C
C XT NCPTS + KORD BREAKPOINT SEQUENCE FOR GENERATION OF BASIS
C WORK(IW2) FUNCTION VALUES.
C
C XC NCPTS COLLOCATION POINTS.
C WORK(IW3)
C
C CHAX NPDE*NCPTS VALUES USED IN ESTIMATING TIME
C CHMAX NPDE*NCPTS INTEGRATION ERRORS.
C
C WORK(IW4)
C
C ERROR NPDE*NCPTS TIME INTEGRATION ERRORS.
C WORK(IW5)
C
C SAVE1 NPDE*NCPTS WORKING STORAGE FOR THE TIME INTEGRATION
C WORK(IW6) METHOD.
C
C SAVE2 NPDE*NCPTS WORKING STORAGE FOR THE TIME INTEGRATION
C WORK(IW7) METHOD.
C
C SAVE3 NPDE*NCPTS WORKING STORAGE FOR THE TIME INTEGRATION
C WORK(IW8) METHOD.
C
C UVAL 3*NPDE WORKING STORAGE FOR VALUES OF U, UX, AND
C WORK(IW9) UX AT ONE POINT.
C
C Y NPDE*NCPTS* CURRENT BASIS FUNCTION COEFFICIENT VALUES
C WORK(IW10) (MAXDER+1) AND THEIR SCALED TIME DERIVATIVES.
C
C DFDU NPDE**2 WORKING STORAGE USED TO COMPUTE THE
C WORK(IW11) JACOBIAN MATRIX.
C
C DFDUX NPDE**2 WORKING STORAGE USED TO COMPUTE THE
C WORK(IW12) JACOBIAN MATRIX.
C
C DFDUXX NPDE**2 WORKING STORAGE USED TO COMPUTE THE
C WORK(IW13) JACOBIAN MATRIX.

```

```

C DBDU ( )
C WORK ( )
C NPDE**2 BOUNDARY CONDITION INFORMATION. (
C DBDUX
C WORK(1W15) BOUNDARY CONDITION INFORMATION.
C DZDT
C WORK(1W16) BOUNDARY CONDITION INFORMATION.
C PW
C WORK(1W17) STORAGE AND PROCESSING OF THE JACOBIAN
C NCPTS (3*ML+1) MATRIX.
C ILEFT NCPTS POINTERS TO BREAKPOINT SEQUENCE FOR
C IWORK GENERATION OF BASIS FUNCTION VALUES.
C IPIV NPDE*NCPTS PIVOT INFORMATION FOR THE LU DECOMPOSED
C IWORK(1W18) JACOBIAN MATRIX PW.
C WHERE...
C NCPTS = KORD*NINT - NCC*(NINT-1)
C ML = NPDE*(KORD+IQAD-1) - 1
C IQAD = 1 IF KORD = 3 AND A NULL BOUNDARY CONDITION EXISTS
C IQAD = 0 OTHERWISE
C MAXDER = 5 UNLESS OTHERWISE SET BY THE USER INTO /OPTION/.
C THE COMMON BLOCK /OPTION/ CONTAINS THE TWO VARIABLES NOGAUS AND
C MAXDER. NOGAUS IS SET .EQ. 0 IN THE BLOCK DATA. IT CAN BE CHANGED
C TO BE SET .EQ. 1 IF THE GAUSS-LEGENDRE COLLOCATION POINTS ARE NOT
C DESIRED WHEN NCC = 2 (SEE ABOVE AND COLNT). MAXDER IS SET
C .EQ. 5 IN THE BLOCK DATA AND ITS VALUE REPRESENTS THE
C MAXIMUM ORDER OF TIME INTEGRATION FORMULA ALLOWED. ITS VALUE
C AFFECTS THE STORAGE REQUIRED IN WORK AND MAY BE CHANGED IF
C DESIRED. SEE COSET FOR RESTRICTIONS. THESE CHANGES MAY BE MADE BY
C THE USER BY ACCESSING /OPTION/ IN HIS CALLING PROGRAM (BEFORE THE
C FIRST CALL TO PDECOL) OR BY CHANGING THE DATA STATEMENT IN
C THE BLOCK DATA.
C-----
C COMMUNICATION
C EACH SUBROUTINE IN THE PACKAGE CONTAINS A COMMUNICATION SUMMARY
C AS INDICATED BELOW.
C PACKAGE ROUTINES CALLED.. EVAL,INITAL,INTERP,STIFB
C USER ROUTINES CALLED.. BNDY
C CALLED BY.. USERS MAIN PROGRAM
C FORTRAN FUNCTIONS USED.. ABS,AMAX1,FLOAT, SORT
C-----
C DIMENSION WORK(1),IWORK(1),XBKPT(1)
C COMMON /GEAR0/ DTUSED,NQUSED,NSTEP,NFE,NJE
C COMMON /GEAR1/ T,DTC,DTMX,DTMX,EPSC,URUND,N,MFC,KFLAG,JSTART
C COMMON /GEAR9/ EPSJ,RO,ML,RU,WM,NM1,NOML,NOW
C COMMON /OPTION/ NOGAUS,MAXDER
C COMMON /SIZES/ NIN,KOR,NC,NPD,NCPTS,NEON,IQAD
C COMMON /ISTART/ IW1,IW2,IW3,IW4,IW5,IW6,IW7,IW8,IW9,IW10,
C * IW11,IW12,IW13,IW14,IW15,IW16,IW17,IW18
C COMMON /COUNT/ LOUT
C IF (INDEX .EQ. 0) GO TO 60
C IF (INDEX .EQ. 2) GO TO 70
C IF (INDEX .EQ. 4) GO TO 80
C IF (INDEX .EQ. 3) GO TO 90
C-----

```

C SEVERAL CHECKS ARE MADE HERE TO DETERMINE IF THE INPUT PARAMETERS
 C HAVE "AL" VALUES. ERROR CHECKS ARE MADE ON INDEX, EPS, (TO-TO-TO)*DT,
 C DT, N, KORD, NCC, NPDE, MF, WHETHER THE BREAKPOINT SEQUENCE
 C STRICTLY INCREASING, AND WHETHER THERE IS SUFFICIENT STORAGE
 C PROVIDED FOR WORK AND IWORK. PROBLEM DEPENDENT PARAMETERS ARE
 C CALCULATED AND PLACED IN COMMON.

```

IERID = -6
IF (INDEX .NE. 1) GO TO 320
IERID = IERID - 1
IF (EPS .LE. 0.) GO TO 320
IERID = IERID - 1
IF ((TO-TO-TO)*DT .GT. 0.) GO TO 320
IERID = IERID - 1
IF (DT .EQ. 0.0) GO TO 320
IERID = IERID - 1
NIN = NINT
IF (NIN .LT. 1) GO TO 320
IERID = IERID - 1
KOR = KORD
IF (KOR .LT. 3 .OR. KOR .GT. 20) GO TO 320
IERID = IERID - 1
NC = NCC
IF (NCC .LT. 2 .OR. NCC .GE. KOR) GO TO 320
IERID = IERID - 1
NPDE = NPDE
NPDE2 = NFO*NPDE
IF (NPDE .LT. 1) GO TO 320
IERID = IERID - 1
IF (MF.NE.22.AND.MF.NE.21.AND.MF.NE.12.AND.MF.NE.11) GO TO 320
IERID = IERID - 1
DO 10 K=1,NIN
  IF (XBKPT(K) .GE. XBKPT(K+1)) GO TO 320
10 CONTINUE
NCPTS = KOR + (NIN - 1) * (KOR - NCC)
NEON = NPDE * NCPTS
ML = (KOR-1)*NPDE - 1
MW = ML
MW = ML + ML + 1
NOW = NEON*MW
IWSAVE = IWORK(1)
IISAVE = IWORK(2)
IW1 = 4*NPDE2 + 1
IW2 = IW1 + 3*KORD*NCPTS
IW3 = IW2 + NCPTS + KORD
IW4 = IW3 + NCPTS
IW5 = IW4 + NEON
IW6 = IW5 + NEON
IW7 = IW6 + NEON
IW8 = IW7 + NEON
IW9 = IW8 + NEON
IW10 = IW9 + 3*NPDE
IW11 = IW10 + NEON*(MAXDER+1)
IW12 = IW11 + NPDE2
IW13 = IW12 + NPDE2
IW14 = IW13 + NPDE2
IW15 = IW14 + NPDE2
IW16 = IW15 + NPDE2
IW17 = IW16 + NPDE2
IW18 = NCPTS + 1
IERID = IERID - 1
IWMSTOR = IW17 + NEON*(3*ML+1) - 1
IISAVE = IW18 + NEON - 1
IF (IWSAVE .LT. IWMSTOR .OR. IISAVE .LT. IWMSTOR) GO TO 335
C PERFORM INITIALIZATION TASKS. IF KORD .EQ. 3 THEN CALCULATE THE BAND-

```

```

C WIDTH OF THE ASSOCIATED MATRIX PROBLEM BY DETERMINING THE TYPE OF
C BOUNDARY CONDITIONS, THEN CHECK FOR SUFFICIENT STORAGE AGAIN. (
-----
      CALL INITAL(KOR,WORK(IW1),WORK(IW6),XBKPT,WORK(IW2),WORK(IW3),
      *      WORK(IW7),WORK(IW8),IWORK)
      IF(IQUAD.NE.0) GO TO 280
      IF(KOR.NE.3) GO TO 40
      CALL EVAL(I,NPDE,WORK(IW6),WORK(IW9),WORK(IW1),IWORK)
      CALL BNDRY(TO,WORK(IW3),WORK(IW9),WORK(IW9+NPDE),WORK(IW14),
      *      WORK(IW15),WORK(IW16),NPDE)
      DO 20 K=1,NPDE
      *      I = K + NPDE*(K-1) - 1
      *      IF(WORK(IW14+I).EQ.0.0.AND.WORK(IW15+I).EQ.0.0)
      *      *      IQUAD = 1
      *      20 CONTINUE
      *      CALL EVAL(NCPTS,NPDE,WORK(IW6),WORK(IW9),WORK(IW1),IWORK)
      *      CALL BNDRY(TO,WORK(IW3+NCPTS-1),WORK(IW9),WORK(IW9+NPDE),
      *      *      WORK(IW14),WORK(IW15),WORK(IW16),NPDE)
      *      DO 30 K=1,NPDE
      *      *      I = K + NPDE*(K-1) - 1
      *      *      IF(WORK(IW14+I).EQ.0.0.AND.WORK(IW15+I).EQ.0.0)
      *      *      *      IQUAD = 1
      *      *      30 CONTINUE
      *      *      ML = ML + IQUAD*NPDE
      *      *      MU = ML
      *      *      MW = ML + ML + 1
      *      *      NM = NEON*MW
      *      40 CONTINUE
      *      INKSTOR = IW17 + NEON*(3*ML+1) - 1
      *      IF ( INKSAVE .LT. INKSTOR ) GO TO 335
      *      -----
      *      C IF INITIAL VALUES OF CMAX OTHER THAN THOSE SET BELOW ARE DESIRED,
      *      C THEY SHOULD BE SET HERE. ALL CMAX(I) MUST BE POSITIVE.
      *      C HAVING PROPER VALUES OF CMAX FOR THE PROBLEM BEING SOLVED IS AS
      *      C IMPORTANT AS CHOOSING EPS (SEE ABOVE), SINCE ERRORS ARE
      *      C MEASURED RELATIVE TO CMAX. IF VALUES FOR DTMIN OR DTMAX, THE
      *      C BOUNDS ON ABS(DT), OTHER THAN THOSE BELOW ARE DESIRED, THEY
      *      C SHOULD BE SET BELOW.
      *      -----
      *      DO 50 I = 1,NEON
      *      *      I1 = I - 1
      *      *      WORK(IW4+I1) = ABS(WORK(IW6+I1))
      *      *      IF (WORK(IW4+I1).LT.1.) WORK(IW4+I1) = 1.
      *      50 WORK(IW10+I1) = WORK(IW6+I1)
      *      *      N = NEON
      *      *      T = TO
      *      *      DTC = DT
      *      *      DTMIN = ABS(DT)
      *      *      DTUSED = 0.
      *      *      EPSC = EPS
      *      *      MFC = MF
      *      *      JSTART = 0
      *      *      EPSJ = SORT(UROUND)
      *      *      NM1 = NEON - 1
      *      *      NOML = NEON*ML
      *      *      NHCUT = 0
      *      *      KFLAG = 0
      *      *      TOUTP = TO
      *      *      IF ( TO.EQ.TOUT ) GO TO 360
      *      60 DTMX = ABS(TOUT-TOUTP)*10.
      *      *      GO TO 140
      *      *      C
      *      *      70 DTMX = ABS(TOUT-TOUTP)*10.
      *      *      IF ((T-TOUT)*DTC.GE.0.) GO TO 340
      *      *      GO TO 150
      *      *      C

```

```

80 IF (T-TOUT)*DTC .GE. 0.) GO TO 300
   IF T = -1
     EPSC = EPS
     KFC = MF
     GO TO 100
C
90 DTMX = DT
100 IF ((T+DTC) .EQ. T) WRITE(LOUT,110)
110 FORMAT(36H WARNING.. T + DT = T ON NEXT STEP.)
C
C TAKE A TIME STEP BY CALLING THE INTEGRATOR.
C
   CALL STIFIB (NEON,WORK(I410),WORK(I44),WORK(I45),WORK(I46),
   * WORK(I47),WORK(I48),WORK(I417),WORK(I418),WORK(I419))
C
   KGO = 1 - KFLAG
   GO TO (120, 160, 220, 260, 280), KGO
C KFLAG = 0, -1, -2, -3, -4
C
120 CONTINUE
C
C NORMAL RETURN FROM INTEGRATOR.
C
C THE WEIGHTS CHAI(I) ARE UPDATED. IF DIFFERENT VALUES ARE DESIRED,
C THEY SHOULD BE SET HERE. A TEST IS MADE FOR EPS BEING TOO SMALL
C FOR THE MACHINE PRECISION.
C
C ANY OTHER TESTS OR CALCULATIONS THAT ARE REQUIRED AFTER EVERY
C STEP SHOULD BE INSERTED HERE.
C
C IF INDEX = 3, SAVE1 IS SET TO THE CURRENT C VALUES ON RETURN.
C IF INDEX = 2, DT IS CONTROLLED TO HIT TOUT (WITHIN ROUNDOFF
C ERROR), AND THEN THE CURRENT C VALUES ARE PUT IN SAVE1 ON RETURN.
C FOR ANY OTHER VALUE OF INDEX, CONTROL RETURNS TO THE INTEGRATOR
C UNLESS TOUT HAS BEEN REACHED. THEN INTERPOLATED VALUES OF C ARE
C COMPUTED AND STORED IN SAVE1 ON RETURN.
C IF INTERPOLATION IS NOT DESIRED, THE CALL TO INTERP SHOULD BE
C REMOVED AND CONTROL TRANSFERRED TO STATEMENT 340 INSTEAD OF 360.
D = 0.
DO 130 I = 1, NEON
  II = I - 1
  AVI = ABS(WORK(I410+II))
  WORK(I44+II) = AMAX1(WORK(I44+II), AVI)
130 D = D + (AVI/WORK(I44+II))*.2
  D = D - (UROUND/EPS)*.2
  IF (D .GT. FLOAT(NEON)) GO TO 240
  IF (INDEX .EQ. 3) GO TO 340
  IF (INDEX .EQ. 2) GO TO 150
140 IF ((T-TOUT)*DTC .LT. 0.) GO TO 100
  CALL INTERP(TOUT,WORK(I410),NEON,WORK(I46))
  GO TO 360
C
150 IF ((T+DTC)-TOUT)*DTC .LE. 0.) GO TO 100
  IF (ABS(T-TOUT) .LE. 100.*UROUND*DTMX) GO TO 340
  IF ((T-TOUT)*DTC .GE. 0.) GO TO 340
  DTC = (TOUT - T)*(1. - 4.*UROUND)
  JSTART = -1
  GO TO 100
C
C ON AN ERROR RETURN FROM INTEGRATOR, AN IMMEDIATE RETURN OCCURS IF
C KFLAG = -2 OR -4, AND RECOVERY ATTEMPTS ARE MADE OTHERWISE.
C TO RECOVER, DT AND DTMX ARE REDUCED BY A FACTOR OF .1 UP TO 10
C TIMES BEFORE GIVING UP.
C
160 WRITE (LOUT,170) T

```

```

170 FORMAT(//35H KFLAG = -1 FROM INTEGRATOR AT T = .E16.8/
* / M ERROR TEST FAILED WITH ABS(DT) = DTMIN/)
180 IF (NHOUT .EQ. 10) GO TO 200
NHOUT = NHOUT + 1
DTMIN = .1*DTMIN
DTC = .1*DTC
*WRITE (LOUT,190) DTC
190 FORMAT(25H DT HAS BEEN REDUCED TO .E16.8.
* 26H AND STEP WILL BE RETRIED//)
JSTART = -1
GO TO 100
C
200 *RITE (LOUT,210)
210 FORMAT(//44H PROBLEM APPEARS UNSOLVABLE WITH GIVEN INPUT//)
GO TO 340
C
220 WRITE (LOUT,230) T,DTC
230 FORMAT(//35H KFLAG = -2 FROM INTEGRATOR AT T = .E16.8,6H DT =
* .E16.8/52H THE REQUESTED ERROR IS SMALLER THAN CAN BE HANDLED//)
GO TO 340
C
240 *RITE (LOUT,250) T
250 FORMAT(//37H INTEGRATION HALTED BY DRIVER AT T = .E16.8/
* 56H EPS TOO SMALL TO BE ATTAINED FOR THE MACHINE PRECISION//)
KFLAG = -2
GO TO 340
C
260 WRITE (LOUT,270) T
270 FORMAT(//35H KFLAG = -3 FROM INTEGRATOR AT T = .E16.8/
* 45H CORRECTOR CONVERGENCE COULD NOT BE ACHIEVED//)
GO TO 180
C
280 WRITE (LOUT,290)
290 FORMAT(//28H SINGULAR MATRIX ENCOUNTERED.
* 35H PROBABLY DUE TO STORAGE OVERWRITES//)
KFLAG = -4
GO TO 340
C
300 WRITE (LOUT,310) T,TOUT,DTC
310 FORMAT(//45H INDEX = -1 ON INPUT WITH (T-TOUT)*DT .GE. 0./
* 4H T = .E16.8,9H TOUT = .E16.8,8H DTC = .E16.8/
* 44H INTERPOLATION WAS DONE AS ON NORMAL RETURN./
* 41H DESIRED PARAMETER CHANGES WERE NOT MADE.)
CALL INTERP(TOUT,WORK(IW10),NEQN,WORK(IW6))
INDEX = -5
RETURN
C
320 *RITE (LOUT,330) IERID
330 FORMAT(//24H ILLEGAL INPUT...INDEX = ,13//)
INDEX = IERID
RETURN
C
335 WRITE (LOUT,336) IWSIOR,IWSAVE,IISTOR,IISAVE
336 FORMAT(//21H INSUFFICIENT STORAGE/24H WORK MUST BE OF LENGTH,
* 110,5X,12HYOU PROVIDED,110/24H IWORK MUST BE OF LENGTH,110,5X,
* 12HYOU PROVIDED,110//)
INDEX = IERID
RETURN
C
340 TOUT = T
DO 350 I = 1,NEQN
I1 = I - 1
350 WORK(IW6+I1) = WORK(IW10+I1)
360 INDEX = KFLAG
TOUTP = TOUT
DT = DTUSED

```



```

IF (KFLAG .NE. 0) DT = DTC
RE" RN
EX
SUBROUTINE VALUES(X,USOL,SCATCH,NDIM1,NDIM2,NPTS,NDERV,WORK)
C-----
C SUBROUTINE VALUES COMPUTES THE SOLUTION U AND THE FIRST NDERV
C DERIVATIVES OF U AT THE NPTS POINTS X AND AT TIME TOUT AND RETURNS
C THEM IN THE ARRAY USOL. THIS ROUTINE MUST BE USED TO OBTAIN
C SOLUTION VALUES SINCE PDECOL DOES NOT RETURN ANY SOLUTION VALUES
C TO THE USER. SEE PDECOL.
C
C THE CALLING PARAMETERS ARE...
C X = AN ARBITRARY VECTOR OF SPATIAL POINTS OF LENGTH NPTS AT
C WHICH THE SOLUTION AND THE FIRST NDERV DERIVATIVE VALUES
C ARE TO BE CALCULATED. IF X .LT. XLEFT ( X .GT. XRIGHT )
C THEN THE PIECEWISE POLYNOMIAL OVER THE LEFTMOST ( RIGHT-
C MOST ) INTERVAL IS EVALUATED TO CALCULATE THE SOLUTION
C VALUES AT THIS UNUSUAL VALUE OF X. SEE PDECOL.
C
C USOL = AN ARRAY WHICH CONTAINS THE SOLUTION AND THE FIRST
C NDERV DERIVATIVES OF THE SOLUTION AT ALL THE POINTS IN
C THE INPUT VECTOR X. IN PARTICULAR, USOL(J,I,K) CONTAINS
C THE VALUE OF THE (K-1)-ST DERIVATIVE OF THE J-TH PDE
C COMPONENT AT THE I-TH POINT OF THE X VECTOR FOR
C J = 1 TO NPDE, I = 1 TO NPTS, AND K = 1 TO NDERV+1.
C
C SCATCH = A USER SUPPLIED WORKING STORAGE ARRAY OF LENGTH AT LEAST
C KORD*(NDERV+1). SEE BELOW AND PDECOL FOR DEFINITIONS OF
C THESE PARAMETERS.
C
C NDIM1 = THE FIRST DIMENSION OF THE OUTPUT ARRAY USOL IN THE CALLING
C PROGRAM. NDIM1 MUST BE .GE. NPDE.
C
C NDIM2 = THE SECOND DIMENSION OF THE OUTPUT ARRAY USOL IN THE
C CALLING PROGRAM. NDIM2 MUST BE .GE. NPTS.
C
C NPTS = THE NUMBER OF POINTS IN THE X VECTOR.
C
C NDERV = THE NUMBER OF DERIVATIVE VALUES OF THE SOLUTION THAT ARE
C TO BE CALCULATED. NDERV SHOULD BE LESS THAN KORD SINCE
C THE KORD-TH DERIVATIVE OF A POLYNOMIAL OF DEGREE KORD-1
C IS EQUAL TO ZERO. SEE PDECOL.
C
C WORK = THE USERS WORKING STORAGE ARRAY WHICH IS USED IN THIS CASE
C TO PROVIDE THE CURRENT BASIS FUNCTION COEFFICIENTS AND THE
C PIECEWISE POLYNOMIAL BREAKPOINT SEQUENCE.
C
C PACKAGE ROUTINES CALLED... BSPLVD,INTERV
C USER ROUTINES CALLED... NONE
C CALLED BY... USERS MAIN PROGRAM
C FORTRAN FUNCTIONS USED... NONE
C
C-----
C DIMENSION USOL(NDIM1,NDIM2,NDERV),X(NPTS),SCATCH(1),WORK(1)
C COMMON /SIZES/ NINT,KORD,NCC,NPDE,NCPDS,NEON,IQUAD
C COMMON /ISTART/ IM1,IM2,IM3,IM4,IM5,IM6,IDUM(12)
C DATA ILEFT/O/, NFLAG/O/
C NDERV1 = NDERV + 1
C DO 20 IPTS=1,NPTS
C CALL INTERV(WORK(IM2),NCPDS,X(IPTS),ILEFT,NFLAG)
C CALL BSPLVD(WORK(IM2),KORD,X(IPTS),ILEFT,SCATCH,NDERV1)
C IK = ILEFT - KORD
C DO 10 M=1,NDERV1
C I1 = (M-1)*KORD
C DO 10 K=1,NPDE
C USOL(K,IPTS,M) = 0.

```

```

DO 10 I=1,KORD
  I2 = (I+K-1)*NPDE + I*6 - 1
  ( USOL(K,IPTS,M) = USOL(K,IPTS,M) + WORK(12+K) = SCF( 1+I1) )
10 CONTINUE
20 CONTINUE
RETURN
END
BLOCK DATA
C-----
C IN THE FOLLOWING DATA STATEMENT, SET..
C LOUT = THE LOGICAL UNIT NUMBER FOR THE OUTPUT OF MESSAGES DURING
C THE INTEGRATION.
C NCGAUS = SET .EQ. 1 IF THE GAUSS-LEGENDRE COLLOCATION POINTS ARE
C NOT DESIRED WHEN NCC = 2 (SEE PDECOL AND COLPNT).
C MAXDER = SET .EQ. 5. ITS VALUE REPRESENTS THE MAXIMUM ORDER OF
C THE TIME INTEGRATION ALLOWED. ITS VALUE AFFECTS THE STORAGE
C REQUIRED IN WORK AND MAY BE CHANGED IF DESIRED
C (SEE COSET FOR RESTRICTIONS).
C UROUND = THE UNIT ROUNDOFF OF THE MACHINE, I.E. THE SMALLEST
C POSITIVE U SUCH THAT 1. + U .NE. 1. ON THE MACHINE.
C-----
COMMON /GEAR1 DUM(S),UROUND,LDUM(4)
COMMON /OPTIC/ NCGAUS,MAXDER
COMMON /IOUNT// LOUT
DATA LOUT,NCGAUS,MAXDER,UROUND/3,0,5,7,1E-15/
END
SUBROUTINE INITIAL(K,A,RHS,X,XT,XC,PW,IPIV,ILEFT)
C-----
C INITIAL IS CALLED ONLY ONCE BY PDECOL TO PERFORM INITIALIZATION TASKS.
C THESE TASKS INCLUDE - 1) DEFINING THE PIECEWISE POLYNOMIAL SPACE
C BREAKPOINT SEQUENCE, 2) CALLING THE SUBROUTINE COLPNT TO DEFINE THE
C REQUIRED COLLOCATION POINTS, 3) DEFINING THE PIECEWISE POLYNOMIAL SPACE
C BASIS FUNCTION VALUES (PLUS FIRST AND SECOND DERIVATIVE VALUES) AT
C THE COLLOCATION POINTS, AND 4) DEFINING THE INITIAL BASIS FUNCTION
C COEFFICIENTS WHICH DETERMINE THE PIECEWISE POLYNOMIAL WHICH
C INTERPOLATES THE USER SUPPLIED (UNIT) INITIAL CONDITION FUNCTION(S)
C AT THE COLLOCATION POINTS.
C-----
C K = ORDER OF PIECEWISE POLYNOMIAL SPACE.
C A = BASIS FUNCTION VALUES GENERATED BY INITIAL.
C RHS = TEMPORARY STORAGE USED TO RETURN INITIAL CONDITION COEFFICIENT
C VALUES.
C XT = USER DEFINED PIECEWISE POLYNOMIAL BREAKPOINTS.
C XC = PIECEWISE POLYNOMIAL BREAKPOINT SEQUENCE GENERATED BY INITIAL.
C PW = COLLOCATION POINTS GENERATED BY INITIAL.
C PM = STORAGE FOR BAND MATRIX USED TO GENERATE INITIAL
C COEFFICIENT VALUES.
C IPIV = PIVOT INFORMATION FOR LINEAR EQUATION SOLVER DECB-SOLB.
C ILEFT = POINTERS TO BREAKPOINT SEQUENCE GENERATED BY INITIAL.
C-----
C PACKAGE ROUTINES CALLED.. BSPLVD,COLPNT,DECB,INTERV,SOLB
C USER ROUTINES CALLED.. UNIT
C CALLED BY... PDECOL
C FORTRAN FUNCTIONS USED.. MAX0,MIN0
C-----
DIMENSION A(K,3,1),RHS(1),X(1),XT(1),XC(1),PW(1),IPIV(1),ILEFT(1)
COMMON /SIZES/ NINT,KORD,NCC,NPDE,NCPTS,NEQN,IER
COMMON /GEAR9/ EPSJ,R0,ML,MU,LDUM(3),NOW
MFLAG = -2
IER = 0
C-----
C SET UP THE PIECEWISE POLYNOMIAL SPACE BREAKPOINT SEQUENCE.
C-----
KRPT = KORD - NCC
DO 10 I=1,KORD
  XT(NCPTS+I) = X(NINT+I)

```

```

10  Y*(I) = X(I)
    CJ  I=2,NINT
    I1 = (I-2)*KRPT + KORD
    DO 20 J=1,KRPT
20  XT(I1+J) = X(I)
C-----
C SET UP COLLOCATION POINTS ARRAY XC.
C-----
    CALL COLPNT(X, XC, XT)
C-----
C GENERATE THE ILEFT ARRAY. STORE THE BASIS FUNCTION VALUES IN THE
C ARRAY A. THE ARRAY A IS DIMENSIONED (KORD,3,NCPTS) AND A(K,J,I)
C CONTAINS THE VALUE OF THE (J-1)-ST DERIVATIVE (J = 1,2,3) OF THE K-TH
C NONZERO BASIS FUNCTION (K = 1, ..., KORD) AT THE I-TH COLLOCATION
C POINT (I = 1, ..., NCPTS). SET UP RHS FOR INTERPOLATING THE INITIAL
C CONDITIONS AT THE COLLOCATION POINTS. SET THE INTERPOLATION MATRIX
C INTO THE Banded MATRIX PM.
C-----
    DO 30 I=1,N04
30  PM(I) = 0.
    DO 40 J=1,NCPTS
    CALL INTERV(XT,NCPTS,XC(I),ILEFT(I),MFLAG)
    CALL BSPLVDIAT,KORD,XC(I),ILEFT(I),A(1,1,1),3)
    I1 = NPDE - (I-1)
    CALL UINIT(XC(I),RHS(I1+1),NPDE)
    ICOL = ILEFT(I) - I - 1
    J1 = MAX(1,I-2-NCPTS)
    J2 = MIN(KORD,KORD+I-2)
    DO 40 J=J1,J2
    J1 = I1 + NEON * (NPDE * (ICOL + J) - 1)
    DO 40 J=J1,NPDE
40  PM(J,J1+1) = A(J,1,1)
C-----
C LU DECOMPOSE THE MATRIX PM.
C-----
    CALL DECB (NEON,NEON,ML,MU,PW,IPIV,IER)
    IF ( IER .NE. 0 ) RETURN
C-----
C SOLVE THE LINEAR SYSTEM PM*Z = RHS. THIS GIVES THE BASIS FUNCTION
C COEFFICIENTS FOR THE INITIAL CONDITIONS.
C-----
    CALL SOLB (NEON,NEON,ML,MU,PW,RHS,IPIV)
    RETURN
END
SUBROUTINE COLPNT(X, XC, XT)
C-----
C COLPNT IS CALLED ONLY ONCE BY INITIAL TO DEFINE THE REQUIRED COLLOCA-
C TION POINTS WHICH ARE TO BE USED WITH THE USER SELECTED PIECEWISE
C POLYNOMIAL SPACE. THE COLLOCATION POINTS ARE CHOSEN SUCH THAT THEY
C ARE EITHER THE POINTS AT WHICH THE PIECEWISE POLYNOMIAL SPACE BASIS
C FUNCTIONS ATTAIN THEIR UNIQUE MAXIMUM VALUES, OR, THE GAUSS-LEGENDRE
C QUADATURE POINTS WITHIN EACH PIECEWISE POLYNOMIAL SPACE SUBINTERVAL,
C DEPENDING UPON THE SPACE BEING USED AND THE DESIRE OF THE USER.
C-----
C X = USER DEFINED PIECEWISE POLYNOMIAL BREAKPOINTS.
C XC = COLLOCATION POINTS DEFINED BY COLPNT.
C XT = PIECEWISE POLYNOMIAL BREAKPOINT SEQUENCE.
C-----
C PACKAGE ROUTINES CALLED.. BSPLVD, INTERV
C USER ROUTINES CALLED.. NONE
C CALLED BY.. INITIAL
C FORTRAN FUNCTIONS USED.. NONE
C-----
    DIMENSION RHO(40),X(1),XC(1),XT(1)
    COMMON /SIZES/ NINT,KORD,NCC,NPDE,NCPTS,NEON,IQUAD
    COMMON /OPTION/ NOGAUS,MAXDER

```

```

DATA ILEFT/O/
C IF THE VARIABLE NOGAUS IN THE COMMON BLOCK /OPTION/ IS SET .EQ.
C THE USE OF THE GAUSS-LEGENDRE POINTS IS PROHIBITED FOR ALL CASES.
C NOGAUS IS CURRENTLY SET .EQ. 0 BY A DATA STATEMENT IN THE BLOCK DATA.
C THE USER MAY CHANGE THIS AS DESIRED.
C-----
      IF ( NCC .NE. 2 .OR. NOGAUS .EQ. 1 ) GO TO 200
C-----
C COMPUTE THE COLLOCATION POINTS TO BE AT THE GAUSS-LEGENDRE POINTS IN
C EACH PIECEWISE POLYNOMIAL SPACE SUBINTERVAL. THE ARRAY RHO IS SET TO
C CONTAIN THE GAUSS-LEGENDRE POINTS FOR THE STANDARD INTERVAL (-1,1).
C-----
      IPTS = KORD - 2
      GO TO (10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170,
      * 180), IPTS
10  RHO(1) = 0.
      GO TO 190
20  RHO(2) = .577350269189626E-00
      RHO(1) = - RHO(2)
      GO TO 190
30  RHO(3) = .774596669241483E-00
      RHO(1) = - RHO(3)
      RHO(2) = 0.
      GO TO 190
40  RHO(3) = .339981043584856E-00
      RHO(2) = - RHO(3)
      RHO(4) = .861136311594053E-00
      RHO(1) = - RHO(4)
      GO TO 190
50  RHO(4) = .538469310105683E-00
      RHO(2) = - RHO(4)
      RHO(5) = .906179845938684E-00
      RHO(1) = - RHO(5)
      RHO(3) = 0.
      GO TO 190
60  RHO(4) = .238619186083197E-00
      RHO(3) = - RHO(4)
      RHO(5) = .661209386466265E-00
      RHO(2) = - RHO(5)
      RHO(6) = .932469514203152E-00
      RHO(1) = - RHO(6)
      GO TO 190
70  RHO(5) = .405845151377397E-00
      RHO(3) = - RHO(5)
      RHO(6) = .74153118559394E-00
      RHO(2) = - RHO(6)
      RHO(7) = .949107912342759E-00
      RHO(1) = - RHO(7)
      RHO(4) = 0.
      GO TO 190
80  RHO(5) = .183434642495650E-00
      RHO(4) = - RHO(5)
      RHO(6) = .525532409916329E-00
      RHO(3) = - RHO(6)
      RHO(7) = .796668477413627E-00
      RHO(2) = - RHO(7)
      RHO(8) = .960289856497536E-00
      RHO(1) = - RHO(8)
      GO TO 190
90  RHO(5) = 0.
      RHO(6) = .324253423403809E-00
      RHO(7) = .613371432700590E-00
      RHO(8) = .83603107328636E-00
      RHO(9) = .968160239507826E-00
      DO 95 I=1,4

```

```

95  RHO(1) = -RHO(10-1)
   GO TO 190
100 RHO( 6) = .118874338981631E-00
    RHO( 7) = .433395394129247E-00
    RHO( 8) = .674039568299024E-00
    RHO( 9) = .865063366888884E-00
    RHO(10) = .973906528517172E-00
    DO 105 I=1,5
      RHO(I) = -RHO(11-I)
105  GO TO 190
110 RHO( 6) = .0
    RHO( 7) = .269543155952315E-00
    RHO( 8) = .519096129206812E-00
    RHO( 9) = .730152005574049E-00
    RHO(10) = .88706259978095E-00
    RHO(11) = .978228658146057E-00
    DO 115 I=1,5
      RHO(I) = -RHO(12-I)
115  GO TO 190
120 RHO( 7) = .125233408511459E-00
    RHO( 8) = .36783198998180E-00
    RHO( 9) = .587317954286617E-00
    RHO(10) = .75990267419435E-00
    RHO(11) = .904117256370475E-00
    RHO(12) = .881560634246719E-00
    DO 125 I=1,6
      RHO(I) = -RHO(13-I)
125  GO TO 190
130 RHO( 7) = .0
    RHO( 8) = .230458315955135E-00
    RHO( 9) = .48492751036447E-00
    RHO(10) = .642349339440340E-00
    RHO(11) = .801578090733310E-00
    RHO(12) = .91759839922978E-00
    RHO(13) = .984183054718588E-00
    DO 135 I=1,6
      RHO(I) = -RHO(14-I)
135  GO TO 190
140 RHO( 8) = .108054948707344E-00
    RHO( 9) = .319112368927890E-00
    RHO(10) = .515248636358154E-00
    RHO(11) = .687292904811685E-00
    RHO(12) = .827201315069765E-00
    RHO(13) = .928434883663574E-00
    RHO(14) = .96628380666812E-00
    DO 145 I=1,7
      RHO(I) = -RHO(15-I)
145  GO TO 190
150 RHO( 8) = .0
    RHO( 9) = .201194093997435E-00
    RHO(10) = .394151347077563E-00
    RHO(11) = .570972172608539E-00
    RHO(12) = .72417731360170E-00
    RHO(13) = .848206583410427E-00
    RHO(14) = .93273392400706E-00
    RHO(15) = .987992518020485E-00
    DO 155 I=1,7
      RHO(I) = -RHO(16-I)
155  GO TO 190
160 RHO( 9) = .950125098376374E-01
    RHO(10) = .281603550779259E-00
    RHO(11) = .45801677657227E-00
    RHO(12) = .61787624402644E-00
    RHO(13) = .755404408355003E-00
    RHO(14) = .865631202387832E-00
    RHO(15) = .944575023073233E-00

```

```

RHO(16) = .959400934981650E-00
DO 65 I=1,8
  J(I) = -RHO(17-I)
165 GO TO 190
170 RHO( 9) = .0
  RHO(10) = .178484181495848E-00
  RHO(11) = .351231763453876E-00
  RHO(12) = .512690537086477E-00
  RHO(13) = .657671159216691E-00
  RHO(14) = .781514003896801E-00
  RHO(15) = .880239153726986E-00
  RHO(16) = .950675521768768E-00
  RHO(17) = .990575475314417E-00
DO 175 I=1,8
  RHO(1) = -RHO(18-I)
175 GO TO 190
180 RHO(10) = .847750130417353E-01
  RHO(11) = .251886225691506E-00
  RHO(12) = .411751161462843E-00
  RHO(13) = .559770831073948E-00
  RHO(14) = .691687043060353E-00
  RHO(15) = .803704958972523E-00
  RHO(16) = .832602466497556E-00
  RHO(17) = .95823949571396E-00
  RHO(18) = .991585168420931E-00
DO 185 I=1,9
  RHO(1) = -RHO(19-I)
185 GO TO 190
C-----
C COMPUTE THE GAUSS-LEGENDRE COLLOCATION POINTS IN EACH SUBINTERVAL.
C-----
190 DO 195 I=1,MINT
  FAC = ( X(I+1) - X(I) ) * .5
  DO 195 J=1,IPTS
    NCOT = IPTS * (I-1) + J + 1
    XC(KNOT) = X(I) + FAC * ( RHO(J) + 1. )
    XC(1) = X(1)
    XC(NCPTS) = X(MINT+1)
    RETURN
195 XC(1) = X(1)
    XC(NCPTS) = X(MINT+1)
    RETURN
C-----
C COMPUTE THE COLLOCATION POINTS TO BE AT THE POINTS WHERE THE BASIS
C FUNCTIONS ATTAIN THEIR MAXIMA. A BISECTION METHOD IS USED TO FIND
C THE POINTS TO MACHINE PRECISION. THIS PROCESS COULD BE SPEEDED UP
C BY USING A SECANT METHOD IF DESIRED.
C-----
200 ITOP = NCPTS - 1
  MFLAG = -2
  XC(1) = X(1)
  XC(NCPTS) = X(MINT+1)
  DO 240 I=2,ITOP
    XOLD = 1.E+20
    XL = XT(1)
    XR = XT(I+KORD)
    XNEW = .5 * (XL + XR)
    IF( XOLD.EQ.XNEW ) GO TO 240
    CALL INTERV(XT,NCPTS,XNEW,ILEFT,MFLAG)
    CALL BSPLVD(XT,KORD,XNEW,ILEFT,RHO,2)
    DO 220 J=1,KORD
      IF( 1.EQ. J + ILEFT - KORD ) GO TO 230
      CONTINUE
      XVAL = RHO(KORD+J)
      IF( XVAL.EQ.0.0 ) XR = XNEW
      IF( XVAL.GT.0.0 ) XL = XNEW
      IF( XVAL.LT.0.0 ) XR = XNEW
      XOLD = XNEW
      GO TO 210
210
220
230
240 XC(I) = XR

```

```

      RETURN
      EV
      SUBROUTINE BSPLVD ( XT, K, X, ILEFT, VNIKK, NDERIV )
      C-----
      C THIS SUBROUTINE IS PART OF THE B-SPLINE PACKAGE FOR THE STABLE
      C EVALUATION OF ANY B-SPLINE BASIS FUNCTION OR DERIVATIVE VALUE.
      C SEE REFERENCE BELOW.
      C
      C CALCULATES THE VALUE AND THE FIRST NDERIV-1 DERIVATIVES OF ALL
      C B-SPLINES WHICH DO NOT VANISH AT X. THE ROUTINE FILLS THE TWO-
      C DIMENSIONAL ARRAY VNIKK(J,IDERIV). J=IDERIV, ..., K WITH NONZERO
      C VALUES OF B-SPLINES OF ORDER K+1-IDERIV. IDERIV=NDERIV, ..., 1, BY
      C REPEATED CALLS TO BSPLVN.
      C
      C XT = PIECEWISE POLYNOMIAL BREAKPOINT SEQUENCE.
      C K = ORDER OF THE PIECEWISE POLYNOMIAL SPACE.
      C X = POINT AT WHICH THE B-SPLINE IS TO BE EVALUATED.
      C ILEFT = POINTER TO THE BREAKPOINT SEQUENCE.
      C VNIKK = TABLE OF B-SPLINE VALUES AND DERIVATIVES.
      C NDERIV = DETERMINES NUMBER OF DERIVATIVES TO BE GENERATED.
      C
      C REFERENCE
      C
      C DEBOOR, C., PACKAGE FOR CALCULATING WITH B-SPLINES, SIAM J.
      C NUMER. ANAL., VOL. 14, NO. 3, JUNE 1977, PP. 441-472.
      C
      C PACKAGE ROUTINES CALLED... BSPLVN
      C USER ROUTINES CALLED... NONE
      C CALLED BY... COLPNT,INITAL,VALUES
      C FORTRAN FUNCTIONS USED... FLOAT,NMAXO
      C-----
      DIMENSION XT(1),VNIKK(K,NDERIV)
      DIMENSION A(20,20)
      KO = K + 1 - NDERIV
      CALL BSPLVN(XT,KO,1,X,ILEFT,VNIKK(NDERIV,NDERIV))
      IF (NDERIV.LE.1) GO TO 120
      IDERIV = NDERIV
      DO 20 I=2,NDERIV
        IDERVM = IDERIV-I
        DO 10 J=IDERIV,K
          VNIKK(J-1,IDERVM) = VNIKK(J,IDERIV)
        IDERIV = IDERVM
        CALL BSPLVN(XT,0.2,X,ILEFT,VNIKK(IDERIV,IDERIV))
      20 CONTINUE
      DO 40 I=1,K
        DO 30 J=1,K
          A(I,J) = 0.
        A(I,1) = 1.
      KMD = K
      DO 110 M=2,NDERIV
        KMD = KMD - 1
        FKMD = FLOAT(KMD)
        I = ILEFT
        J = K
      50 JN1 = J-1
        IPKMD = I + KMD
        DIFF = XT(IPKMD) - XT(I)
        IF (JN1.EQ.0) GO TO 80
        IF (DIFF.EQ.0.) GO TO 70
        DO 60 L=1,J
          A(L,J) = (A(L,J) - A(L,J-1))/DIFF*FKMD
        J = JN1
        I = I - 1
      GO TO 50
      80 IF (DIFF.EQ.0.) GO TO 90
        A(1,1) = A(1,1)/DIFF*FKMD
      90

```

```

90 DO 110 I=1,K
   V = 0.
   ( LOW = MAXO(I,M)
     DO 100 J=JLOW,K
100   V = A(I,J)*VNIKK(J,M) + V
110   VNIKK(I,M) = V
120 RETURN
END
SUBROUTINE BSPLVN ( XT, JHIGH, INDEX, X, ILEFT, VNIKK )
-----
C THIS SUBROUTINE IS PART OF THE B-SPLINE PACKAGE FOR THE STABLE
C EVALUATION OF ANY B-SPLINE BASIS FUNCTION OR DERIVATIVE VALUE.
C SEE REFERENCE BELOW.
C
C CALCULATES THE VALUE OF ALL POSSIBLY NONZERO B-SPLINES AT THE
C POINT X OF ORDER MAX(JHIGH,(J+1)(INDEX-1)) FOR THE BREAKPOINT SEQ-
C UENCE XT. ASSUMING THAT XT(ILEFT) .LE. X .LE. XT(ILEFT+1), THE ROUT-
C INE RETURNS THE B-SPLINE VALUES IN THE ONE DIMENSIONAL ARRAY VNIKK.
C
C FOR DEFINITIONS OF CALLING ARGUMENTS SEE ABOVE AND BSPLVD.
C
C REFERENCE
C
C DEBOOR, C., PACKAGE FOR CALCULATING WITH B-SPLINES, SIAM J.
C NUMER. ANAL., VOL. 14, NO. 3, JUNE 1977, PP. 441-472.
C
C PACKAGE ROUTINES CALLED.. NONE
C USER ROUTINES CALLED.. NONE
C CALLED BY.. BSPLVD
C FORTRAN FUNCTIONS USED.. NONE
C
-----
DIMENSION XT(1),VNIKK(1)
DIMENSION DELTAM(20),DELTAP(20)
DATA J/1,DELTAM/20*0.E-00,DELTAP/20*0.E-00/
GO TO (10,20),INDEX
10 J = 1
   VNIKK(1) = 1.
   IF (J .GE. JHIGH) GO TO 40
   IF (J .LE. ILEFT) GO
20 IPU = ILEFT+J
   DELTAP(J) = XT(IPU) - X
   IMJP1 = ILEFT-J+1
   DELTAM(J) = X - XT(IMJP1)
   VMPREV = 0.
   JPI = J+1
   DO 30 L=1,J
     JPIML = JPI-L
     VM = VNIKK(L)/(DELTAP(L) + DELTAM(JPIML))
     VNIKK(L) = VM*DELTAP(L) + VMPREV
     VMPREV = VM*DELTAM(JPIML)
30   VNIKK(JPI) = VMPREV
     J = JPI
   IF (J .LT. JHIGH) GO TO 20
40 RETURN
END
SUBROUTINE INTERV ( XT, LXT, X, ILEFT, MFLAG )
-----
C THIS SUBROUTINE IS PART OF THE B-SPLINE PACKAGE FOR THE STABLE
C EVALUATION OF ANY B-SPLINE BASIS FUNCTION OR DERIVATIVE VALUE.
C SEE REFERENCE BELOW.
C
C COMPUTES LARGEST ILEFT IN (1,LXT) SUCH THAT XT(ILEFT) .LE. X. THE
C PROGRAM STARTS THE SEARCH FOR ILEFT WITH THE VALUE OF ILEFT THAT WAS
C RETURNED AT THE PREVIOUS CALL (AND WAS SAVED IN THE LOCAL VARIABLE
C ILO) TO MINIMIZE THE WORK IN THE COMMON CASE THAT THE VALUE OF X ON
C THIS CALL IS CLOSE TO THE VALUE OF X ON THE PREVIOUS CALL. SHOULD
C THIS ASSUMPTION NOT BE VALID, THEN THE PROGRAM LOCATES ILO AND IHI

```



```

C SUCH I XT(ILO) .LE. X .LT. XT(IMI) AND, ONCE THEY ARE FOUND, IS
C BISECTION TO FIND THE CORRECT VALUE FOR ILEFT. MFLAG IS AN ERROR FLAG.
C
C FOR DEFINITIONS OF CALLING ARGUMENTS SEE ABOVE AND BSPLVD.
C
C REFERENCE
C
C DEBOR, C., PACKAGE FOR CALCULATING WITH B-SPLINES, SIAM J.
C NUMER. ANAL., VOL. 14, NO. 3, JUNE 1977, PP. 441-472.
C
C PACKAGE ROUTINES CALLED... NONE
C USER ROUTINES CALLED... NONE
C CALLED BY... COLPNT,INITIAL,VALUES
C FORTRAN FUNCTIONS USED... NONE
C-----
C DIMENSION XT(LXT)
C IF(MFLAG.EQ.-2) ILO = 1
C IMI = ILO + 1
C IF (IMI .LT. LXT) GO TO 20
C IF (X .GE. XT(LXT)) GO TO 110
C IF (LXT .LE. 1) GO TO 90
C ILO = LXT - 1
C GO TO 21
C 20 IF (X .GE. XT(IMI)) GO TO 40
C 21 IF (X .GE. XT(ILO)) GO TO 100
C-----
C NOW X .LT. XT(IMI). FIND LOWER BOUND.
C-----
C 30 ISTEP = 1
C 31 IMI = ILO
C ILO = IMI - ISTEP
C IF (ILO .LE. 1) GO TO 35
C IF (X .GE. XT(ILO)) GO TO 50
C ISTEP = ISTEP*2
C GO TO 31
C 35 ILO = 1
C IF (X .LT. XT(1)) GO TO 90
C GO TO 50
C-----
C NOW X .GE. XT(ILO). FIND UPPER BOUND.
C-----
C 40 ISTEP = 1
C 41 ILO = IMI
C IMI = ILO + ISTEP
C IF (IMI .GE. LXT) GO TO 45
C IF (X .LT. XT(IMI)) GO TO 50
C ISTEP = ISTEP*2
C GO TO 41
C 45 IF (X .GE. XT(LXT)) GO TO 110
C IMI = LXT
C-----
C NOW XT(ILO) .LE. X .LT. XT(IMI). NARROW THE INTERVAL.
C-----
C 50 MIDDLE = (ILO + IMI)/2
C IF (MIDDLE .EQ. ILO) GO TO 100
C-----
C NOTE.. IT IS ASSUMED THAT MIDDLE = ILO IN CASE IMI = ILO+1.
C-----
C IF (X .LT. XT(MIDDLE)) GO TO 53
C ILO = MIDDLE
C GO TO 50
C 53 IMI = MIDDLE
C GO TO 50
C-----
C SET OUTPUT AND RETURN.
C-----

```

```

90 KFLAG = -1
   IL = 1
   IF (N) = 1
100 MF = 0
   ILEFT = ILO
   RETURN
110 KFLAG = 1
   ILEFT = LAT
   RETURN
   END
SUBROUTINE STIFIB (NO, Y, YMAX, ERROR, SAVE1, SAVE2, SAVE3,
                  PM, IDIV, WORK, IWORK)
C-----
C STIFIB PERFORMS ONE STEP OF THE INTEGRATION OF AN INITIAL VALUE
C PROBLEM FOR A SYSTEM OF ORDINARY DIFFERENTIAL EQUATIONS OF THE FORM,
C  $A(Y,T) \cdot (DY/DT) = G(Y,T)$ , WHERE  $Y = (Y(1), Y(2), \dots, Y(N))$ .
C STIFIB IS FOR USE WHEN THE MATRICES A AND DG/DY HAVE BANDED OR NEARLY
C BANDED FORM. THE DEPENDENCE OF A(Y,T) ON Y IS ASSUMED TO BE WEAK.
C-----
C REFERENCE
C HINDMARSH, A.C., PRELIMINARY DOCUMENTATION OF GEARIB, SOLUTION
C OF IMPLICIT SYSTEMS OF ORDINARY DIFFERENTIAL EQUATIONS WITH
C BANDED JACOBIANS, LAWRENCE LIVERMORE LAB, UCID-30130, FEBRUARY
C 1976.
C COMMUNICATION WITH STIFIB IS DONE WITH THE FOLLOWING VARIABLES..
C Y AN NO BY LMAX ARRAY CONTAINING THE DEPENDENT VARIABLES
C AND THEIR SCALED DERIVATIVES. LMAX IS 13 FOR THE ADAMS
C METHODS AND 6 FOR THE GEAR METHODS. LMAX - 1 = MAXDER
C IS THE MAXIMUM ORDER AVAILABLE. SEE SUBROUTINE COSET.
C Y(1..N-1) CONTAINS THE J-TH DERIVATIVE OF Y(1), SCALED BY
C  $H^{(J+1)/FACTORIAL(J)}$  (J = 0, 1, ..., NO).
C NO A CONSTANT INTEGER .GE. N, USED FOR DIMENSIONING PURPOSES.
C T THE INDEPENDENT VARIABLE. T IS UPDATED ON EACH STEP TAKEN.
C H THE STEP SIZE TO BE ATTEMPTED ON THE NEXT STEP.
C H IS ALTERED BY THE ERROR CONTROL ALGORITHM DURING THE
C PROBLEM. H CAN BE EITHER POSITIVE OR NEGATIVE, BUT ITS
C SIGN MUST REMAIN CONSTANT THROUGHOUT THE PROBLEM.
C HMIN THE MINIMUM AND MAXIMUM ABSOLUTE VALUE OF THE STEP SIZE
C TO BE USED FOR THE STEP. THESE MAY BE CHANGED AT ANY
C TIME, BUT WILL NOT TAKE EFFECT UNTIL THE NEXT H CHANGE.
C EPS THE RELATIVE ERROR BOUND. SEE DESCRIPTION IN PDECOL.
C N THE UNIT ROUNDOFF OF THE MACHINE.
C MF THE NUMBER OF FIRST-ORDER DIFFERENTIAL EQUATIONS.
C KFLAG THE METHOD FLAG. SEE DESCRIPTION IN PDECOL.
C A COMPLETION CODE WITH THE FOLLOWING MEANINGS..
C 0 THE STEP WAS SUCCESSFUL.
C -1 THE REQUESTED ERROR COULD NOT BE ACHIEVED
C WITH  $ABS(H) \geq HMIN$ .
C -2 THE REQUESTED ERROR IS SMALLER THAN CAN
C BE HANDLED FOR THIS PROBLEM.
C -3 CORRECTOR CONVERGENCE COULD NOT BE
C ACHIEVED FOR  $ABS(H) \geq HMIN$ .
C -4 SINGULAR A-MATRIX ENCOUNTERED.
C ON A RETURN WITH KFLAG NEGATIVE, THE VALUES OF T AND
C THE Y ARRAY ARE AS OF THE BEGINNING OF THE LAST
C STEP, AND H IS THE LAST STEP SIZE ATTEMPTED.
C JUSTART AN INTEGER USED ON INPUT AND OUTPUT.
C ON INPUT, IT HAS THE FOLLOWING VALUES AND MEANINGS..
C 0 PERFORM THE FIRST STEP.
C .GT.0 TAKE A NEW STEP CONTINUING FROM THE LAST.
C .LT.0 TAKE THE NEXT STEP WITH A NEW VALUE OF
C H, EPS, N, AND/OR MF.
C ON EXIT, JUSTART IS NO, THE CURRENT ORDER OF THE METHOD.

```

```

C YMA1 AN ARRAY OF N ELEMENTS WITH WHICH THE ESTIMATED LOCAL
C ERRORS IN Y ARE COMPARED.
C ERROR AN ARRAY OF N ELEMENTS. ERROR(1)/TQ(2) IS THE ESTIMATED
C ONE-STEP ERROR IN Y(1).
C SAVE1,SAVE2,SAVE3 THREE WORKING STORAGE ARRAYS, EACH OF LENGTH N.
C PW A BLOCK OF LOCATIONS USED FOR THE CHORD ITERATION
C MATRIX SEE DESCRIPTION IN PDECOL.
C IPIV AN INTEGER ARRAY OF LENGTH N FOR PIVOT INFORMATION.
C ML,MLU THE LOWER AND UPPER HALF BANDWIDTHS, RESPECTIVELY, OF
C THE CHORD ITERATION MATRIX. SEE DESCRIPTION IN PDECOL.
C WORK,WORK WORKING ARRAYS WHICH ARE USED TO PASS APPROPRIATE
C ARRAYS TO OTHER SUBROUTINES.
C
C PACKAGE ROUTINES CALLED.. COSET,DIFFUN,PSETIB,RES,SOLB
C USER ROUTINES CALLED.. NONE
C CALLED BY.. PDECOL
C FORTRAN FUNCTIONS USED.. ABS,AMAX1,AMIN1,FLOAT
C
C DIMENSION Y(N0,1),YMAX(N0),ERROR(N0),SAVE1(N0),SAVE2(N0),
C SAVE3(N0),P4(1),IPIV(1),KORR(1),IWORK(1)
C COMMON /SIZES/ NINT,KORD,ACC,NPOE,SCRIPTS,NEON,LOUAD
C COMMON /ISTART/ IWT,IW2,IW3,IW4,IW5,IW6,IW7,IW8,IW9,IW10,IW11,
C IW12,IW13,IW14,IW15,IW16,IW17,IW18
C COMMON /GEAR1/ T,M,HMIN,HMAX,EPS,UROUND,N,MF,AFLAG,JSTART
C COMMON /GEAR2/ EPSJ,RO,ML,MM,NMT,NOML,NOW
C COMMON /GEAR3/ HUSED,NQUSED,NSTEP,NFE,NJE
C COMMON /OPTION/ NOGAUS,MAXDER
C DIMENSION EL(13),TQ(4)
C DATA EL(2)/1.,.OLDLO/1.,.TQ(1)/0.,.IER/0/
C KFLAG = 0
C TOLD = T
C IF (JSTART .GT. 0) GO TO 200
C IF (JSTART .NE. 0) GO TO 120
C
C ON THE FIRST CALL, THE ORDER IS SET TO 1 AND THE INITIAL YDOT IS
C CALCULATED. RMAX IS THE MAXIMUM RATIO BY WHICH H CAN BE INCREASED
C IN A SINGLE STEP. IT IS INITIALLY 1.E4 TO COMPENSATE FOR THE SMALL
C INITIAL H, BUT THEN IS NORMALLY EQUAL TO 10. IF A FAILURE
C OCCURS (IN CORRECTOR CONVERGENCE OR ERROR TEST), RMAX IS SET AT 2
C FOR THE NEXT INCREASE.
C
C H0 = 1
C IER = 0
C CALL DIFFUN (N, T, Y, SAVE1, IER, PW, IPIV, WORK, IWORK)
C IF (IER .NE. 0) GO TO 685
C DO 110 I = 1, N
C 110 Y(I,2) = H*SAVE1(I)
C METH = MF/10
C MITER = MF - 10*METH
C L = 2
C IDOUB = 3
C RMAX = 1.E+04
C RC = 0.
C CRATE = 1.
C EPSOLD = EPS
C HOLD = H
C MOLD = MF
C NOLD = N
C NSTEP = 0
C NSTEPJ = 0
C NFE = 0
C NJE = 1
C IRET = 3
C GO TO 130
C
C IF THE CALLING PROGRAMMER WOULD WANT TO CALL TO SET

```

C THE COEFFICIENTS OF THE METHOD. IF THE CALLER HAS CHANGED
C N, E, OR METH, THE CONSTANTS E, EDN, EUP, AND BND MUST BE RE-
C E IS A COMPARISON FOR ERRORS OF THE CURRENT ORDER NO. EUP IS
C TO TEST FOR INCREASING THE ORDER, EDN FOR DECREASING THE ORDER.
C BND IS USED TO TEST FOR CONVERGENCE OF THE CORRECTOR ITERATES.
C IF THE CALLER HAS CHANGED M, Y MUST BE RESCALED.
C IF M OR METH HAS BEEN CHANGED, IDOUB IS RESET TO L + 1 TO PREVENT
C FURTHER CHANGES IN H FOR THAT MANY STEPS.

120 IF (MF.EQ.MFOLD) GO TO 150

MEO = METH,
MIO = MITER,
METH = MF/10
MITER = MF - 10*METH
MFOLD = MF
IF (MITER.NE.MIO) IWEVAL = MITER
IF (METH.EQ.MEO) GO TO 150
IDOUB = L + 1
IRET = 1

130 CALL COSET (METH, NQ, EL, TQ)
RC = RC-EL(1)/OLDLO
OLDLO = EL(1)

140 FN = FDATE(N)
EDN = FN-(TQ(1)-EPS)**2
E = FN-(TQ(2)-EPS)**2
EUP = FN-(TQ(3)-EPS)**2
BND = FN-(TQ(4)-EPS)**2
GO TO (160, 170, 200), IRET

150 IF ((EPS.EQ.EPSOLD).AND.(N.EQ.NOLD)) GO TO 160
EPSOLD = EPS
NOLD = N
IRET = 1

GO TO 140
160 IF (H.EQ.HOLD) GO TO 200
RH = H/HOLD
H = HOLD
IREDO = 3

GO TO 175
170 RH = AMAX1(RH,RHMIN/ABS(H))
175 RH = AMIN1(RH,RHMAX/ABS(H),RMAX)
R1 = 1.
DO 180 J = 2, L

R1 = R1+RH
DO 180 I = 1, N
180 Y(I,J) = Y(I,J)*R1
H = H+RH
RC = RC+RH
IDOUB = L + 1

IF (IREDO.EQ.0) GO TO 690

C THIS SECTION COMPUTES THE PREDICTED VALUES BY EFFECTIVELY
C MULTIPLYING THE Y ARRAY BY THE PASCAL TRIANGLE MATRIX.
C RC IS THE RATIO OF NEW TO OLD VALUES OF THE COEFFICIENT M*EL(1).
C WHEN RC DIFFERS FROM 1 BY MORE THAN 30 PERCENT, OR THE CALLER HAS
C CHANGED MITER, IWEVAL IS SET TO MITER TO FORCE PW TO BE UPDATED.
C IN ANY CASE, PW IS UPDATED AT LEAST EVERY 40-TH STEP.
C PW IS THE CHORD ITERATION MATRIX A - M*EL(1)*(DG/DY).

200 IF (ABS(RC-1.) .GT. 0.3) IWEVAL = MITER
IF (NSTEP.GE.NSTEP+40) IWEVAL = MITER
T = T + H
DO 210 J1 = 1, NQ
DO 210 J2 = J1, NQ
J = (NQ + J1) - J2
DO 210 I = 1, N

```

210      Y(I,J) = Y(I,J) + Y(I,J+1)
C-----
C UP TO J-CORRECTOR ITERATIONS ARE TAKEN. A CONVERGENCE TEST IS
C MADE ON THE R.M.S. NORM OF EACH CORRECTION, USING BND, WHICH
C IS DEPENDENT ON EPS. THE SUM OF THE CORRECTIONS IS ACCUMULATED
C IN THE VECTOR ERROR(I). THE Y ARRAY IS NOT ALTERED IN THE CORRECTOR
C LOOP. THE UPDATED Y VECTOR IS STORED TEMPORARILY IN SAVE1.
C THE UPDATED -H*YDOT IS STORED IN SAVE2.
C-----
220 DO 230 I = 1,N
    SAVE2(I) = Y(I,2)
230   ERROR(I) = 0.
    M = 0
    CALL RES (T, H, Y, SAVE2, SAVE3, NPDE, NCPTS, WORK(IW1), IWORK,
    * WORK, WORK(IW14), WORK(IW15), WORK(IW16), WORK(IW3), WORK(IW9))
    NFE = NFE + 1
    IF (IWEVAL .LE. 0) GO TO 350
C-----
C IF INDICATED, THE MATRIX PW IS REEVALUATED BEFORE STARTING THE
C CORRECTOR ITERATION. IWEVAL IS SET TO 0 AS AN INDICATOR
C THAT THIS HAS BEEN DONE. PW IS COMPUTED AND PROCESSED IN PSETIB.
C-----
    IWEVAL = 0
    RC = 1.
    NFE = NFE + 1
    NSTEPJ = NSTEP
    CON = -H*EL(1)
    CALL PSETIB (Y, PW, NO, CON, WITER, IER, WORK(IW1), IWORK,
    * WORK(IW3), WORK(IW9), SAVE2, IP1V, YMAX, WORK(IW11), WORK(IW12),
    * WORK(IW13), WORK(IW16), WORK(IW14), WORK(IW15), WORK, NPDE)
    IF (IER .NE. 0) GO TO 420
C-----
C COMPUTE THE CORRECTOR ERROR, R SUB M, AND SOLVE THE LINEAR SYSTEM
C WITH THAT AS RIGHT-HAND SIDE AND PW AS COEFFICIENT MATRIX.
C USING THE LU DECOMPOSITION OF PW.
C-----
350 CALL SOLB (NO, N, ML, MU, PW, SAVE3, IP1V)
370 D = 0.
DO 380 I = 1,N
    ERROR(I) = ERROR(I) + SAVE3(I)
D = D + (SAVE3(I)/YMAX(I))*2
SAVE1(I) = Y(I,1) + EL(1)*ERROR(I)
380   SAVE2(I) = Y(I,2) + ERROR(I)
C-----
C TEST FOR CONVERGENCE. IF M.GT.0, AN ESTIMATE OF THE CONVERGENCE
C RATE CONSTANT IS STORED IN CRATE, AND THIS IS USED IN THE TEST.
C-----
400 IF (M .NE. 0) CRATE = AMAX1(.9*CRATE,D/D1)
    IF (10*AMINT(1.E0/2.*CRATE)) .LE. BND) GO TO 450
    D1 = D
    M = M + 1
    IF (M .EQ. 3) GO TO 410
    CALL RES(T, H, SAVE1, SAVE2, SAVE3, NPDE, NCPTS, WORK(IW1), IWORK,
    * WORK, WORK(IW14), WORK(IW15), WORK(IW16), WORK(IW3), WORK(IW9))
    GO TO 350
C-----
C THE CORRECTOR ITERATION FAILED TO CONVERGE IN 3 TRIES.
C IF THE MATRIX PW IS NOT UP TO DATE, IT IS REEVALUATED FOR THE
C NEXT TRY. OTHERWISE THE Y ARRAY IS RETRACTED TO ITS VALUES
C BEFORE PREDICTION, AND H IS REDUCED, IF POSSIBLE. IF NOT, A
C NO-CONVERGENCE EXIT IS TAKEN.
C-----
410 NFE = NFE + 2
    IF (IWEVAL .EQ. -1) GO TO 440
420 I = TOLD
    RMAX = 2.

```

```

DO 430 J1 = 1,NQ
(
  430 J2 = J1,NQ
    J = (NQ + J1) - J2
    DO 430 I = 1,N
      Y(I,J) = Y(I,J) - Y(I,J+1)
    430 IF (ABS(H) .LE. HMIN*1.00001) GO TO 680
    RH = .25
    IREDO = 1
    GO TO 170
  440 IAEVAL = MITER
    GO TO 220
)
C THE CORRECTOR HAS CONVERGED. IAEVAL IS SET TO -1 TO SIGNAL
C THAT WE MAY NEED UPDATING ON SUBSEQUENT STEPS. THE ERROR TEST
C IS MADE AND CONTROL PASSES TO STATEMENT 500 IF IT FAILS.
C
450 IAEVAL = -1
NFE = NFE + M
D = 0.
DO 460 I = 1,N
  460 D = D + (ERROR(I)/YMAX(I))**2
  IF (D .GT. E) GO TO 500
C
C AFTER A SUCCESSFUL STEP, UPDATE THE Y ARRAY.
C CONSIDER CHANGING H IF IDOUB = 1. OTHERWISE DECREASE IDOUB BY 1.
C IF IDOUB IS THEN 1 AND NO .LT. MAXDER THEN ERROR IS SAVED FOR
C USE IN A POSSIBLE ORDER INCREASE ON THE NEXT STEP.
C IF A CHANGE IN H IS CONSIDERED, AN INCREASE OR DECREASE IN ORDER
C BY ONE IS CONSIDERED ALSO. A CHANGE IN H IS MADE ONLY IF IT IS BY A
C FACTOR OF AT LEAST 1.1. IF NOT, IDOUB IS SET TO 10 TO PREVENT
C TESTING FOR THAT MANY STEPS.
C
KFLAG = 0
IREDO = 0
NSTEP = NSTEP + 1
HUSED = H
HUSED = NO
DO 470 J = 1,L
  DO 470 I = 1,N
    Y(I,J) = Y(I,J) + EL(J)*ERROR(I)
  470 IF (IDOUB .EQ. 1) GO TO 520
  IDOUB = IDOUB - 1
  IF (IDOUB .GT. 1) GO TO 700
  IF (L .EQ. LMAX) GO TO 700
  DO 490 I = 1,N
    490 Y(I,LMAX) = ERROR(I)
  GO TO 700
C
C THE ERROR TEST FAILED. KFLAG KEEPS TRACK OF MULTIPLE FAILURES.
C RESTORE T AND THE Y ARRAY TO THEIR PREVIOUS VALUES, AND PREPARE
C TO TRY THE STEP AGAIN. COMPUTE THE OPTIMUM STEP SIZE FOR THIS OR
C ONE LOWER ORDER.
C
500 KFLAG = KFLAG + 1
T = TOLD
DO 510 J1 = 1,NQ
  DO 510 J2 = J1,NQ
    J = (NQ + J1) - J2
    DO 510 I = 1,N
      Y(I,J) = Y(I,J) - Y(I,J+1)
    510 RMAX = 2.
    IF (ABS(H) .LE. HMIN*1.00001) GO TO 660
    IF (KFLAG .LE. -3) GO TO 640
    IREDO = 2
    PR3 = 1.E+20
    GO TO 540

```

```

C-----
C REGARDS OF THE SUCCESS OR FAILURE OF THE STEP, FACTORS
C PR1, PR2, AND PR3 ARE COMPUTED, BY WHICH H COULD BE DIVIDED
C AT ORDER NO - 1, ORDER NO, OR ORDER NO + 1, RESPECTIVELY.
C IN THE CASE OF FAILURE, PR3 = 1.E20 TO AVOID AN ORDER INCREASE.
C THE SMALLEST OF THESE IS DETERMINED AND THE NEW ORDER CHOSEN
C ACCORDINGLY. IF THE ORDER IS TO BE INCREASED, WE COMPUTE ONE
C ADDITIONAL SCALED DERIVATIVE.
C-----
520 PR3 = 1.E+20
   IF (L.EQ. LMAX) GO TO 540
   D1 = 0.
   DO 530 I = 1,N
     D1 = D1 + ((ERROR(I) - Y(I,LMAX))/YMAX(I))**2
   EN3 = .5/ FLOAT(L+1)
   PR3 = ((D1/EUP)**EN3)*1.4 + 1.4E-06
540 EN2 = .5/ FLOAT(L)
   PR2 = ((D1/E)**EN2)*1.2 + 1.2E-06
   PR1 = 1.E+20
   IF (NQ.EQ. 1) GO TO 560
   D = 0.
   DO 550 I = 1,N
     D = D + (Y(I,L)/YMAX(I))**2
   EN1 = .5/ FLOAT(NQ)
   PR1 = ((D/EDN)**EN1)*1.3 + 1.3E-06
560 IF (PR2.LE. PR3) GO TO 570
   IF (PR3.LT. PR1) GO TO 590
   GO TO 580
570 IF (PR2.GT. PR1) GO TO 580
   NEWQ = NQ
   RH = 1./PR2
   GO TO 620
580 NEWQ = NQ - 1
   RH = 1./PR1
   GO TO 620
590 NEWQ = L
   RH = 1./PR3
   IF (RH.LT. 1.1) GO TO 610
   DO 600 I = 1,N
     Y(I,NEWQ+1) = ERROR(I)*EL(L)/ FLOAT(L)
   GO TO 630
610 ICGUB = 10
   GO TO 700
620 IF ((KFLAG.EQ. 0) .AND. (RH.LT. 1.1)) GO TO 610
C-----
C IF THERE IS A CHANGE OF ORDER, RESET NQ, L, AND THE COEFFICIENTS.
C IN ANY CASE H IS RESET ACCORDING TO RH AND THE Y ARRAY IS RESCALED.
C THEN EXIT FROM 690 IF THE STEP WAS OK, OR REDO THE STEP OTHERWISE.
C-----
   IF (NEWQ.EQ. NQ) GO TO 170
630 NQ = NEWQ
   L = NQ + 1
   IRET = 2
   GO TO 130
C-----
C CONTROL REACHES THIS SECTION IF 3 OR MORE FAILURES HAVE OCCURRED.
C IT IS ASSUMED THAT THE DERIVATIVES THAT HAVE ACCUMULATED IN THE
C Y ARRAY HAVE ERRORS OF THE WRONG ORDER. HENCE THE FIRST
C DERIVATIVE IS RECOMPUTED, AND THE ORDER IS SET TO 1. THEN
C H IS REDUCED BY A FACTOR OF 10, AND THE STEP IS RETRIED.
C AFTER A TOTAL OF 7 FAILURES, AN EXIT IS TAKEN WITH KFLAG = -2.
C-----
640 IF (KFLAG.EQ. -7) GO TO 670
   RH = .1
   RH = AMAX1(HMIN/ ABS(H),RH)
   U = -U

```

```

IF = 0
C DIFFUN (N, T, Y, SAVE1, IER, PW, IPIV, WORK, IWORK) (
IF ( IER.NE. 0 ) GO TO 685
NJE = NJE + 1
DO 650 I = 1,N
650 Y(I,2) = H*SAVE1(I)
I*EVAL = MITER
IDUB = 10
IF (NQ.EQ. 1) GO TO 200
NQ = 1
L = 2
IRET = 3
GO TO 130
-----
C ALL RETURNS ARE MADE THROUGH THIS SECTION. H IS SAVED IN HOLD
C TO ALLOW THE CALLER TO CHANGE H ON THE NEXT STEP.
C
660 KFLAG = -1
GO TO 700
670 KFLAG = -2
GO TO 700
680 KFLAG = -3
GO TO 700
685 KFLAG = -4
GO TO 700
690 RNAX = 10.
700 HOLD = H
JSTART = NQ
RETURN
END
*
SUBROUTINE GFUN (T,C,UOOT,NPDE,NCPTS,A,BC,DBDU,DBDUX,DZDT,
XC,UVAL,ILEFT)
-----
C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL.
C
C SUBROUTINE GFUN COMPUTES THE FUNCTION UOOT=G(C,T). THE RIGHT-
C HAND SIDE OF THE SEMI-DISCRETE APPROXIMATION TO THE ORIGINAL
C SYSTEM OF PARTIAL DIFFERENTIAL EQUATIONS AND UPDATES THE BOUNDARY
C CONDITION INFORMATION.
C
C PACKAGE ROUTINES CALLED.. EVAL
C USER ROUTINES CALLED.. BNDRY,F
C CALLED BY.. DIFFUN,PSETIB,RES
C FORTRAN FUNCTIONS USED.. NONE
C
DIMENSION C(NPDE,NCPTS),UOOT(NPDE,NCPTS)
DIMENSION A(1),BC(NPDE,NPDE,4),XC(1),UVAL(NPDE,3),ILEFT(1)
DIMENSION DZDT(NPDE),DBDU(NPDE,NPDE),DBDUX(NPDE,NPDE)
COMMON /SIZES/ NINT,KORD,IDUM(4),IQDAD
DO 10 K=1,4
DO 10 J=1,NPDE
DO 10 I=1,NPDE
BC(I,J,K) = 0.0
10 CONTINUE
C
C UPDATE THE LEFT BOUNDARY VALUES. SAVE LEFT BOUNDARY CONDITION
C INFORMATION IN THE FIRST 2*NPDE*NPDE LOCATIONS OF BC.
C
C NOTE.. UVAL(K,1) = U(K), UVAL(K,2) = UX(K), AND UVAL(K,3) = UXX(K).
C
CALL EVAL(1,NPDE,C,UVAL,A,ILEFT)
CALL BNDRY(T,XC(1),UVAL,UVAL(1,2),DBDU,DBDUX,DZDT,NPDE)
CALL F(T,XC(1),UVAL,UVAL(1,2),UVAL(1,3),UOOT,NPDE)
ILIM = KORD + 2
DO 30 K=1,NPDE
BC(K,K,1) = 1.

```



```

      DBDU(K,K) .EQ. 0.0 .AND. DBDUX(K,K) .EQ. 0.0 ) GO ( 30
      DO 20 J=1,NPDE
      BC(K,J,2) = A(ILIM) * DBDUX(K,J)
      BC(K,J,1) = DBDU(K,J) - BC(K,J,2)
20  CONTINUE
30  CONTINUE
C MAIN LOOP TO FORM RIGHT SIDE OF ODES AT THE COLLOCATION POINTS.
C-----
      ILIM = NCPTS - 1
      DO 40 I=2,ILIM
      CALL EVAL(I,NPDE,C,UVAL,A,ILEFT)
      CALL F(T,XC(I),UVAL,UVAL(1,2),UVAL(1,3),YDOT(1,1),NPDE)
40  CONTINUE
C-----
C UPDATE THE RIGHT BOUNDARY VALUES. SAVE THE RIGHT BOUNDARY CONDITION
C INFORMATION IN THE LAST 2*NPDE-NPDE LOCATIONS IN BC.
C-----
      CALL EVAL(NCPTS,NPDE,C,UVAL,A,ILEFT)
      CALL F(T,XC(NCPTS),UVAL,UVAL(1,2),UVAL(1,3),YDOT(1,NCPTS),NPDE)
      CALL BNDY(T,AC(NCPTS),UVAL,UVAL(1,2),DBDU,DBDUX,DZDT,NPDE)
      ILIM = NCPTS - 3 * KORD - KORD - 1
      DO 60 K=1,NPDE
      BC(K,K,4) = 1.
      IF( DBDU(K,K) .EQ. 0.0 .AND. DBDUX(K,K) .EQ. 0.0 ) GO TO 60
      YDOT(K,NCPTS) = DZDT(K)
      DO 50 J=1,NPDE
      BC(K,J,3) = A(ILIM) * DBDUX(K,J)
      BC(K,J,4) = DBDU(K,J) - BC(K,J,3)
50  CONTINUE
60  CONTINUE
      RETURN
      END
      SUBROUTINE EVAL(ICPT,NPDE,C,UVAL,A,ILEFT)
C-----
C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL.
C-----
C SUBROUTINE EVAL EVALUATES U(K), UX(K), AND UXX(K), K=1 TO NPDE,
C AT THE COLLOCATION POINT WITH INDEX ICPT USING THE VALUES OF
C THE BASIS FUNCTION COEFFICIENTS IN C AND THE BASIS FUNCTION VALUES
C STORED IN A. THE RESULTS ARE STORED IN UVAL AS FOLLOWS..
C UVAL(K,1) = U(K), UVAL(K,2) = UX(K), AND UVAL(K,3) = UXX(K).
C-----
C PACKAGE ROUTINES CALLED.. NONE
C USER ROUTINES CALLED.. NONE
C CALLED BY.. GFUN,PDECOL,PSETIB
C FORTRAN FUNCTIONS USED.. NONE
C-----
      DIMENSION C(NPDE,1),UVAL(NPDE,3),A(1),ILEFT(1)
      COMMON /SIZES/ NINT,KORD,IDUM(5)
      IK = ILEFT(ICPT) - KORD
      IC = 3*KORD-(ICPT-1)
      DO 10 M=1,3
      ICC = IC + KORD*(M-1)
      DO 10 J=1,NPDE
      UVAL(J,M) = 0.
      DO 10 I=1,KORD
      UVAL(J,M) = UVAL(J,M) + C(J,I+IK)*A(I+ICC)
10  CONTINUE
      RETURN
      END
      SUBROUTINE DIFFUN (N, T, Y, YDOT, IER, PW, IPIV, WORK, IWORK)
C-----
C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL.
C-----

```

```

C THIS ROUTINE COMPUTES YDOT = A(Y,T) + G(Y,T) BY USE OF
C THE ROUTINES GFUN, ADDA, DECB, AND SOLB.
C
C PACKAGE ROUTINES CALLED.. ADDA,DECB,GFUN,SOLB
C USER ROUTINES CALLED.. NONE
C CALLED BY.. STIFIB
C FORTRAN FUNCTIONS USED.. NONE
C-----
      DIMENSION Y(N),YDOT(N),PA(1),IPIV(1),WORK(1),IWORK(1)
      COMMON /GEAR9/ EPSJ,R0,ML,MU,M1,NM1,NOML,NOW
      COMMON /SIZES/ NINT,KORD,NPDE,NPDE,NCPYS,NEQN,IQUAD
      COMMON /ISTART/ IWI,IW2,IW3,IW4,IW5,IW6,IW7,IW8
      CALL GFUN (T, Y, YDOT, NPDE, NCPYS, WORK(IW1), WORK(IW14),
      *          WORK(IW15), WORK(IW16), WORK(IW3), WORK(IW9), IWORK)
      DO 10 I = 1,NOM
      *
      *          PM(I) = 0.
      *
      *          NO = NM1 + 1
      *          CALL ADDA (PA, NO, WORK(IW1), IWORK, WORK, NPDE)
      *          CALL DECB (NO, N, ML, MU, MU, PA, IPIV, IER)
      *          IF ( IER .NE. 0 ) RETURN
      *          CALL SOLB (NO, N, ML, MU, MU, YDOT, IPIV)
      *          RETURN
      *          END
      *
      *          SUBROUTINE ADDA(PA,NO,A,LEFT,BC,NPDE)
      *
      *          CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL AND STIFIB.
      *
      *          SUBROUTINE ADDA ADDS THE MATRIX A TO THE MATRIX STORED IN PA IN
      *          BAND FORM. PA IS STORED BY DIAGONALS WITH THE LOWERMOST DIAGONAL
      *          STORED IN THE FIRST COLUMN OF THE ARRAY.
      *
      *          PACKAGE ROUTINES CALLED.. NONE
      *          USER ROUTINES CALLED.. NONE
      *          CALLED BY.. DIFFUN,PSETIB
      *          FORTRAN FUNCTIONS USED.. NONE
      *
      *          DIMENSION PA(N0,1),A(1),LEFT(1),BC(NPDE,NPDE,4)
      *          COMMON /SIZES/ NINT,KORD,NCC,NPD,NCPYS,NEQN,IQUAD
      *
      *          C ADD THE BOUNDARY CONDITION PORTIONS OF THE A MATRIX TO PA ( THE FIRST
      *          C AND LAST BLOCK ROWS).
      *
      *          ICOL = (LEFT(1) + IQUAD - 1) * NPDE
      *          DO 10 I=1,NPDE
      *             IBOT = NEQN - NPDE + I
      *             DO 10 J=1,NPDE
      *                INO = ICOL + J - I
      *                PA(I,INO) = PA(I,INO) + BC(I,J,1)
      *                PA(I,INO+NPDE) = PA(I,INO+NPDE) + BC(I,J,2)
      *                PA(IBOT,INO-NPDE) = PA(IBOT,INO-NPDE) + BC(I,J,3)
      *                PA(IBOT,INO) = PA(IBOT,INO) + BC(I,J,4)
      *
      *          10 CONTINUE
      *
      *          C UPDATE THE REMAINING ROWS OF PA BY ADDING THE APPROPRIATE VALUES
      *          C IN A TO PA.
      *
      *          INO = NCPYS - 1
      *          DO 20 I=2,INO
      *             I1 = (I-1) * NPDE
      *             I2 = (I-1) * KORD + 3
      *             ICOL = ILEFT(1) - I + IQUAD - 1
      *             DO 20 J=1,KORD
      *                J1 = (ICOL+J) * NPDE
      *                J2 = I2 + J
      *                DO 20 JJ=1,NPDE

```

```

20  IF (I+J) = PM(I+J,J) + PM(I+J,J) + A(J2)
    END
    SUBROUTINE RES(T,H,C,V,R,NPDE,NCPTS,A,ILEFT,BC,DBDU,DBDUX,DZDT,
    XC,UVAL)
    C-----
    C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL.
    C-----
    C SUBROUTINE RES COMPUTES THE RESIDUAL VECTOR R = H*(C.T) - A(C.T)*V
    C WHERE H IS THE CURRENT TIME STEP SIZE, G IS A VECTOR, A IS A
    C MATRIX, V IS A VECTOR, AND T IS THE CURRENT TIME.
    C-----
    C PACKAGE ROUTINES CALLED.. GFUN
    C USER ROUTINES CALLED.. NONE
    C CALLED BY.. STIFIB
    C FORTRAN FUNCTIONS USED.. NONE
    C-----
    DIMENSION C(NPDE,NCPTS),R(NPDE,1),V(NPDE,1)
    DIMENSION A(1),ILEFT(1),BC(NPDE,NPDE,4),XC(1),UVAL(1)
    DIMENSION DBD(NPDE,NPDE),DBDUX(NPDE),DZDT(NPDE)
    COMMON /SIZES/ NINT,KORD,NCC,LDUM(3),IQDAD
    C-----
    C FORM G(C.T) AND STORE IN R.
    C-----
    CALL GFUN(T,C,R,NPDE,NCPTS,A,BC,DBDU,DBDUX,DZDT,XC,UVAL,ILEFT)
    C-----
    C FORM THE FIRST AND LAST BLOCK ROWS OF THE RESIDUAL VECTOR
    C WHICH ARE DEPENDENT ON THE BOUNDARY CONDITIONS.
    C-----
    ILM = NCPTS - 1
    DO 20 I=1,NPDE
        SUM1 = 0.0
        SUM2 = 0.0
        DO 10 J=1,NPDE
            SUM1 = SUM1 + BC(I,J,1) * V(J,1) + BC(I,J,2) * V(J,2)
            SUM2 = SUM2 + BC(I,J,3) * V(J,ILIM) + BC(I,J,4) * V(J,NCPTS)
        10 CONTINUE
        R(I,1) = H * R(I,1) - SUM1
        R(I,NCPTS) = H * R(I,NCPTS) - SUM2
    20 CONTINUE
    C-----
    C FORM THE REMAINING COMPONENTS OF THE RESIDUAL VECTOR.
    C-----
    DO 50 ICPTS=2,ILM
        I2 = (ICPTS-1) * KORD + 3
        ICOL = ILEFT(ICPTS) - KORD
        DO 40 JJ=1,NPDE
            SUM1 = 0.
            DO 30 J=1,KORD
                SUM1 = SUM1 + A(I2+J) * V(JJ,ICOL+J)
            30 CONTINUE
            R(JJ,ICPTS) = H*R(JJ,ICPTS) - SUM1
        40 CONTINUE
    50 CONTINUE
    RETURN
    END
    SUBROUTINE PSETIB (C, PM, NO, CON, MITER, IER, A, ILEFT, XC, UVAL,
    SAVE2, IP1V, CMAX, DFDU, DFDUX, DFDUX, DZDT, DBDU, DBDUX, BC, NPDE)
    C-----
    C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL AND STIFIB.
    C-----
    C PSETIB IS CALLED BY STIFIB TO COMPUTE AND PROCESS THE MATRIX
    C PW = A - H*EL(1)*(DG/DC), WHERE A AND DG/DC ARE TREATED IN BAND
    C FORM. DG/DC IS COMPUTED, EITHER WITH THE AID OF THE USER-SUPPLIED
    C ROUTINE DERIVF IF MITER = 1, OR BY FINITE DIFFERENCING WITH THE AID
    C OF THE PACKAGE-SUPPLIED ROUTINE DIFF IF MITER = 2. FINALLY,

```

```

C PW IS SUBJECTED TO LU DECOMPOSITION IN PREPARATION FOR LATER
C SOLUTION OF LINEAR SYSTEMS WITH PW AS COEFFICIENT MATRIX.
C SEE SUBROUTINES DECB AND SOLS.

C IN ADDITION TO VARIABLES DESCRIBED PREVIOUSLY, COMMUNICATION
C WITH PSETIB USES THE FOLLOWING..
C EPSJ = SORT(UROUND), USED IN THE NUMERICAL JACOBIAN INCREMENTS.
C MW = ML + MU + 1.
C NM1 = NO - 1.
C NOIL = NO - ML.
C NOX = NO - MM.

C PACKAGE ROUTINES CALLED.. ADDA,DECB,DIFF, EVAL,GFUN
C USER ROUTINES CALLED.. BNDY,DERIVS
C CALLED BY.. STIFIB
C FORTRAN FUNCTIONS USED.. ABS,FLOAT,MAXO,MINO,SORT

C-----
      DIMENSION PW(ND,1),C(1),CMAX(1)
      DIMENSION A(1),ILEFT(1),BC(1),XC(1),UVAL(NPDE,3),SAVE2(1),IPIV(1)
      DIMENSION OFOU(NPDE,NPDE),DFDUX(NPDE,NPDE),DFDUXX(NPDE,NPDE)
      DIMENSION DZDT(NPDE),DBDU(NPDE,NPDE),DBDUX(NPDE,NPDE)
      COMMON /SIZES/ NINT,KORD,NCC,NPD,NCPTS,NEQN,IQUAD
      COMMON /GEAR1/ T,H,DUMMY(3),UROUND,N,IDUMMY(3)
      COMMON /GEARS/ EPSJ,RO,ML,MU,NM1,NOML,NOW
      DO 10 I=1,NOW
10    PW(I,1) = 0.
      IF (MITER.EQ.1) GO TO 25
      CALL GFUN (T,C,SAVE2,NPDE,NCPTS,A,BC,DBDU,DBDUX,DZDT,XC,
      * UVAL,ILEFT)
      D = 0.
      DO 20 I = 1,N
20    D = D + SAVE2(I)**2
      RO = ABS(H)* SORT(D/FLOAT(ND))*1.E+03*UROUND
C-----
C COMPUTE BLOCK ROWS OF JACOBIAN.
C-----
25    DO 30 I=1,NCPTS
      I1 = (I-1)*NPDE
      I2 = (I-1)*KORD+3
      CALL EVAL(I,NPDE,C,UVAL,A,ILEFT)
      IF (MITER.EQ.1)
      * CALL DERIVF(T,XC(1),UVAL,UVAL(1,2),UVAL(1,3),
      * DFDU,DFDUX,DFDUXX,NPDE)
      IF (MITER.EQ.2)
      * CALL DIFF(T,XC(1),I,UVAL,UVAL(1,2),UVAL(1,3),
      * DFDU,DFDUX,DFDUXX,NPDE,CMAX,SAVE2)
      ICOL = ILEFT(I) - 1 + IQUAD - 1
      KLOW = MAXO(1,I2-NCPTS)
      KUP = MINO(KORD,KORD+1-2)
      DO 30 KBLK=KLOW,KUP
      * J1 = (ICOL+KBLK)*NPDE
      * J2 = I2 + KBLK
      * J3 = J2 + KORD
      * J4 = J3 + KORD
      DO 30 L=1,NPDE
      * PW((I1+K,J1-K+L) + DFDU(K,L)*A(J2) + DFDUX(K,L)*A(J3)
      * + DFDUXX(K,L)*A(J4))
30    CONTINUE

C-----
C MODIFY THE LAST AND THE FIRST BLOCK ROWS FOR THE BOUNDARY CONDITIONS.
C CURRENT INFORMATION FOR THE RIGHT BOUNDARY CONDITION IS ALREADY IN
C THE ARRAYS DBDU, DBDUX AS A RESULT OF A PREVIOUS CALL TO GFUN.
C-----
      IROW = NEQN - NPDE
      DO 50 K=1,NPDE

```

```

      IM = IROW + 1
      DBDU(K,K) .EQ. 0.0 .AND. DBDUX(K,K) .EQ. 0.0) GO TO 1
      DO 40 J=1,NW
        PM(IROW,J) = 0.0
      CONTINUE
50 CONTINUE
      CALL EVAL(1,NPDE,C,UVAL,A,ILEFT)
      CALL ENGRY(T,XC(1),UVAL,UVAL(1,2),DBDU,DBDUX,DZDT,NPDE)
      DO 70 K=1,NPDE
        IF(DBDU(K,K) .EQ. 0.0 .AND. DBDUX(K,K) .EQ. 0.0) GO TO 70
        DO 60 J=1,NW
          PM(K,J) = 0.0
        CONTINUE
70 CONTINUE
      DO 80 I = 1,NOW
        PM(I,1) = PM(I,1)*CON
      CONTINUE
C ADD MATRIX A(C,T) TO PM.
C
C CALL ADDA (PM, NO, A, ILEFT, BC, NPDE)
C
C DO LU DECOMPOSITION ON PM.
C
C CALL DECB (NO, N, ML, MU, PM, IPIV, IER)
      RETURN
      END
      SUBROUTINE DIFFF(T,X,IPT,U,UX,UXX,DFDU,DFDUX,DFDUXX,NPDE,CMAX,
      SAVE2)
C CALLING ARGUMENTS ARE DEFINED BELOW AND IN PDECOL.
C
C SUBROUTINE DIFFF IS USED IF MITER=2 TO PROVIDE FINITE DIFFERENCE
C APPROXIMATIONS FOR THE PARTIAL DERIVATIVES OF THE K-TH USER DEFINED
C FUNCTION IN THE F ROUTINE WITH RESPECT TO THE VARIABLES U, UX, AND
C UXX. THESE PARTIALS WITH RESPECT TO U, UX, AND UXX ARE COMPUTED,
C STORED, AND RETURNED IN THE NPDE BY NPDE ARRAYS DFDU, DFDUX, AND
C DFDUXX, RESPECTIVELY, AT COLLOCATION POINT NUMBER IPT.
C
C PACKAGE ROUTINES CALLED.. NONE
C USER ROUTINES CALLED.. F
C CALLED BY.. PSETIB
C FORTRAN FUNCTIONS USED.. AMAX1
C
      DIMENSION U(NPDE),UX(NPDE),UXX(NPDE),DFDU(NPDE,NPDE),
      * DFDUX(NPDE,NPDE),DFDUXX(NPDE,NPDE),CMAX(1),SAVE2(1)
      COMMON /GEARS/ EPSJ,RO,ML,MU,NW,NMT,NML,NOW
      ID = (IPT-1) * NPDE
      DO 40 J=1,NPDE
        UJ = U(J)
        R = EPSJ * CMAX(J)
        R = AMAX1(R,RO)
        U(J) = U(J) + R
        RINV = 1. / R
        CALL FIT(X,U,UX,UXX,DFDU(1,J),NPDE)
        DO 10 I=1,NPDE
          DFDU(I,J) = ( DFDU(I,J) - SAVE2(I+ID) ) * RINV
          UJ = UJ
          UXJ = UX(J)
          UXXJ = UXX(J) + R
          CALL FIT(X,U,UX,UXX,DFDUX(1,J),NPDE)
          DO 20 I=1,NPDE
            DFDUX(I,J) = ( DFDUX(I,J) - SAVE2(I+ID) ) * RINV
            UJ = UJ
            UXXJ = UXXJ
            UXX(J) = UXX(J) + R
            CALL FIT(X,U,UX,UXX,DFDUXX(1,J),NPDE)

```

```

DO 30 I=1,NPDE
30 ( YDUXX(I,J) = ( DFDUXX(I,J) - SAVE2(I+10) ) * RINV (
40 CONTINUE
RETURN
END
SUBROUTINE INTERP (TOUT, Y, NO, Y0)
C-----
C CALL'G ARGUMENTS ARE DEFINED BELOW AND IN STIFB
C
C SUBROUTINE INTERP COMPUTES INTERPOLATED VALUES OF THE DEPENDENT
C VARIABLE Y AND STORES THEM IN Y0. THE INTERPOLATION IS TO THE
C POINT T = TOUT, AND USES THE NORDSTROM HISTORY ARRAY Y, AS FOLLOWS..
C
C NO
C Y0(I) = SUM Y(I,J+1)*S**J.
C J=0
C
C WHERE S = -(T-TOUT)/H.
C
C PACKAGE ROUTINES CALLED.. NONE
C USER ROUTINES CALLED.. NONE
C CALLED BY.. PDECOL
C FORTRAN FUNCTIONS USED.. NONE
C-----
C
C DIMENSION Y0(NO),Y(NO,1)
C COMMON /GEAR/ T,H,DUMMY(4),N,TDUMMY(2),JSTART
C DO 10 I = 1,N
10 Y0(I) = Y(I,1)
L = JSTART + 1
S = (TOUT - T)/H
S1 = 1.
DO 30 J = 2,L
S1 = S1*S
DO 20 I = 1,N
20 Y0(I) = Y0(I) + S1*Y(I,J)
30 CONTINUE
RETURN
END
SUBROUTINE COSET (METH, NO, EL, TQ)
C-----
C COSET IS CALLED BY THE INTEGRATOR AND SETS COEFFICIENTS USED THERE.
C THE VECTOR EL, OF LENGTH NO + 1, DETERMINES THE BASIC METHOD.
C THE VECTOR TQ, OF LENGTH 4, IS INVOLVED IN ADJUSTING THE STEP SIZE
C IN RELATION TO TRUNCATION ERROR. ITS VALUES ARE GIVEN BY THE
C PERTST ARRAY.
C
C THE VECTORS EL AND TQ DEPEND ON METH AND NO.
C THE MAXIMUM ORDER, MAXDER, OF THE METHODS AVAILABLE IS CURRENTLY
C 12 FOR THE ADAMS METHODS AND 5 FOR THE BDF METHODS. MAXDER DEFAULTS
C TO 5 UNLESS THE USER SETS MAXDER TO SOME OTHER LEGITIMATE VALUE
C THROUGH THE COMMON BLOCK /OPTION/. SEE PDECOL FOR ADDITIONAL DETAILS.
C LMAX = MAXDER + 1 IS THE NUMBER OF COLUMNS IN THE Y ARRAY (SEE STIFB
C AND THE VARIABLE C, Y, OR WORK(IW10) IN PDECOL.
C
C THE COEFFICIENTS IN PERTST NEED BE GIVEN TO ONLY ABOUT
C ONE PERCENT ACCURACY. THE ORDER IN WHICH THE GROUPS APPEAR BELOW
C IS.. COEFFICIENTS FOR ORDER NO - 1. COEFFICIENTS FOR ORDER NO.
C COEFFICIENTS FOR ORDER NO + 1. WITHIN EACH GROUP ARE THE
C COEFFICIENTS FOR THE ADAMS METHODS, FOLLOWED BY THOSE FOR THE
C BDF METHODS.
C
C REFERENCE
C GEAR, C.W., NUMERICAL INITIAL VALUE PROBLEMS IN ORDINARY
C DIFFERENTIAL EQUATIONS, PRENTICE-HALL, ENGLEWOOD CLIFFS,
C N. J., 1971.

```

```

C PACKING ROUTINES CALLED.. NONE
C USER ROUTINES CALLED.. NONE
C CALLED BY.. STIFB
C FORTRAN FUNCTIONS USED.. FCAT
C
C DIMENSION PERST(12,2,3),EL(13),TO(4)
C DATA PERST / 1,1,2,3,1,1,2,3,158,07407,.01391,.002182,
C * .0002945,.00033492,.000003692,.000003524,
C * 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
C * 2,12,124,37,89,53,33,70,08,87,97,106,9,
C * 126,7,147,4,168,8,191,0,
C * 2,0,4,5,7,33,10,42,13,7,1,1,1,1,1,1,1,1,1,1,
C * 12,0,24,0,37,89,53,33,70,08,87,97,106,9,
C * 126,7,147,4,168,8,191,0,1,
C * 3,0,6,0,9,167,12,5,1,1,1,1,1,1,1,1,1,1,1,1, /
C
C GO TO (1,2),METH
C 1 GO TO 101,102,103,104,105,106,107,108,109,110,111,112),NO
C 2 GO TO 1201,122,203,204,255),NO
C
C THE FOLLOWING COEFFICIENTS SHOULD BE DEFINED TO MACHINE ACCURACY.
C FOR A GIVEN ORDER NO, THEY CAN BE CALCULATED BY USE OF THE
C GENERATING POLYNOMIAL  $L(T)$ , WHOSE COEFFICIENTS ARE  $EL(1)$ ..
C  $L(T) = EL(1) + EL(2)*T + \dots + EL(NO+1)*T^{NO}$ .
C FOR THE IMPLICIT ADAMS METHODS,  $L(T)$  IS GIVEN BY
C  $OL/DY = (T+1)*(T+2)* \dots *(T+NO-1)/K$ ,  $L(-1) = 0$ ,
C WHERE  $K = FACTORIAL(NO-1)$ .
C FOR THE BDF METHODS,
C  $L(T) = (T+1)*(T+2)* \dots *(T+NO)/K$ ,
C WHERE  $K = FACTORIAL(NO)*(1 + 1/2 + \dots + 1/NO)$ .
C
C THE ORDER IN WHICH THE GROUPS APPEAR BELOW IS..
C IMPLICIT ADAMS METHODS OF ORDERS 1 TO 12,
C BDF METHODS OF ORDERS 1 TO 5.
C
101 EL(1) = 1.0E-00
GO TO 900
102 EL(1) = 0.5E-00
EL(3) = 0.5E-00
GO TO 900
103 EL(1) = 4.1666666666667E-01
EL(3) = 0.75E-00
EL(4) = 1.6666666666667E-01
GO TO 900
104 EL(1) = 0.375E-00
EL(3) = 9.1666666666667E-01
EL(4) = 3.3333333333333E-01
EL(5) = 4.1666666666667E-02
GO TO 900
105 EL(1) = 3.4861111111111E-01
EL(3) = 1.0416666666667E-00
EL(4) = 4.8611111111111E-01
EL(5) = 1.0416666666667E-01
EL(6) = 8.3333333333333E-03
GO TO 900
106 EL(1) = 3.2986111111111E-01
EL(3) = 1.1416666666667E-00
EL(4) = 0.625E-00
EL(5) = 1.7708333333333E-01
EL(6) = 0.025E-00
EL(7) = 1.3888888888889E-03
GO TO 900
107 EL(1) = 3.1559193121693E-01
EL(3) = 1.225E-00
EL(4) = 7.5185185185185E-01
EL(5) = 2.5520833333333E-01

```

EL(1) = 4.881111111111111E-02
 EL(2) = 4.881111111111111E-03
 EL(3) = 1.9841269841270E-04
 GO TO 900
 108
 EL(1) = 3.0422453703704E-01
 EL(2) = 1.2664285714286E-00
 EL(3) = 8.6851851851852E-01
 EL(4) = 3.3576388888889E-01
 EL(5) = 7.7777777777778E-02
 EL(6) = 1.0644148148148E-02
 EL(7) = 7.9365079365079E-04
 EL(8) = 2.4801587301587E-05
 GO TO 900
 109
 EL(1) = 2.9486800044092E-01
 EL(2) = 1.3589285714286E-00
 EL(3) = 9.7655121280423E-01
 EL(4) = 0.4171875E-00
 EL(5) = 1.1135416666667E-01
 EL(6) = 0.01875E-00
 EL(7) = 1.9345238095238E-03
 EL(8) = 1.1160714285714E-04
 EL(9) = 2.7557319223986E-06
 GO TO 900
 110
 EL(1) = 2.8697544642857E-01
 EL(2) = 1.414841269841E-00
 EL(3) = 1.072156084658E-00
 EL(4) = 4.9856701940035E-01
 EL(5) = 0.1484375E-00
 EL(6) = 2.9060570987654E-02
 EL(7) = 3.7202380952381E-03
 EL(8) = 2.9968584656085E-04
 EL(9) = 1.3778653611993E-05
 EL(10) = 2.7557319223986E-07
 GO TO 900
 111
 EL(1) = 2.801895964394E-01
 EL(2) = 1.4644841269841E-00
 EL(3) = 1.171514502646E-00
 EL(4) = 5.7935619003527E-01
 EL(5) = 1.8832286155203E-01
 EL(6) = 4.1430362654321E-02
 EL(7) = 6.2111441798942E-03
 EL(8) = 6.2520667989418E-04
 EL(9) = 4.0417401528513E-05
 EL(10) = 1.5156525573193E-06
 EL(11) = 2.5052108385442E-08
 GO TO 900
 112
 EL(1) = 2.7426554003160E-01
 EL(2) = 1.5099386724387E-00
 EL(3) = 1.2602711640212E-00
 EL(4) = 6.5923418209877E-01
 EL(5) = 2.3045800264550E-01
 EL(6) = 5.5637248105233E-02
 EL(7) = 9.4394841269841E-03
 EL(8) = 1.1192745669312E-03
 EL(9) = 9.0939153439153E-05
 EL(10) = 4.8225308641975E-06
 EL(11) = 1.5031265031265E-07
 EL(12) = 2.0876756987868E-09
 GO TO 900
 201
 EL(1) = 1.0E-00
 GO TO 900
 202
 EL(1) = 6.6666666666667E-01
 EL(3) = 3.3333333333333E-01
 GO TO 900
 203
 EL(1) = 5.4545454545455E-01
 EL(3) = EL(1)


```

204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000

```

```

32 LR = ML
D( ) NR = 1.N1
  IF (LR .NE. N) LR = LR + 1
  MX = NR
  XM = ABS(B(NR,1))
  IF (ML .EQ. 0) GO TO 42
  DO 40 I = NP,LR
    IF (ABS(B(I,1)) .LE. XM) GO TO 40
  MX = I
  XM = ABS(B(I,1))
  CONTINUE
40 IPIV(NR) = MX
42 IF (MX .EQ. NR) GO TO 60
  DO 50 I = 1,LL
    XX = B(NR,I)
    B(NR,I) = B(MX,I)
    B(MX,I) = XX
  IF (XM .EQ. 0.) GO TO 100
  B(NR,1) = 1./XM
  IF (ML .EQ. 0) GO TO 90
  XM = -B(NR,1)
  KK = MIN0(N-NR,LL-1)
  DO 80 I = NP,LR
    J = LL + I - NR
    XX = B(I,1)*XM
    B(NR,J) = XX
  DO 70 II = 1,KK
    B(II,II) = B(II,II+1) + XX*B(NR,II+1)
  B(II,LL) = 0.
70 CONTINUE
80 B(II,LL) = 0.
90 CONTINUE
92 NR = N
  IF (B(N,1) .EQ. 0.) GO TO 100
  BIN(1) = 1./B(N,1)
  RETURN
100 IER = NR
  RETURN
END

```

SUBROUTINE SOLB (NDIM, N, ML, MU, B, Y, IPIV)

C SUBROUTINES DECB AND SOLB FORM A TWO SUBROUTINE PACKAGE FOR THE
C DIRECT SOLUTION OF A SYSTEM OF LINEAR EQUATIONS IN WHICH THE
C COEFFICIENT MATRIX IS REAL AND BANDED.

C
C SOLUTION OF $A \cdot X = C$ GIVEN LU DECOMPOSITION OF A FROM DECB.
C Y = RIGHT-HAND VECTOR C, OF LENGTH N, ON INPUT,
C = SOLUTION VECTOR X ON OUTPUT.

C ALL THE ARGUMENTS ARE INPUT ARGUMENTS.
C THE OUTPUT ARGUMENT IS Y.

C PACKAGE ROUTINES CALLED... NONE
C USER ROUTINES CALLED... NONE
C CALLED BY... DIFFUN,INITAL,STIFIB
C FORTRAN FUNCTIONS USED... MINO

DIMENSION B(NDIM,1),Y(N),IPIV(N)

IF (N .EQ. 1) GO TO 60

N1 = N - 1

LL = ML + MU + 1

IF (ML .EQ. 0) GO TO 32

DO 30 NR = 1,N1

IF (IPIV(NR) .EQ. NR) GO TO 10

J = IPIV(NR)

XX = Y(NR)

Y(NR) = Y(J)

```

      ( J ) = XX
10      . = MINO(N-NR,ML)
      DO 20 I = 1,KK
20      Y(NR+I) = Y(NR+I) + Y(NR)*B(NR,LL+I)
30      CONTINUE
32      LL = LL + 1
34      Y(N) = Y(N)*B(N,1)
      KK = 0
      DO 50 NB = 1,N1
      NR = N - NB
      IF (KK .NE. LL) KK = KK + 1
      DP = 0.
      IF (LL .EQ. 0) GO TO 50
      DO 40 I = 1,KK
40      DP = DP + B(NR,I+1)*Y(NR+I)
50      Y(NR) = (Y(NR) - DP)*B(NR,1)
60      Y(1) = Y(1)*B(1,1)
      RETURN
      END
C      PROGRAM EXAMP:(UGOUT,TAPE3=UGOUT)
      COMMON /ENDPT/ XLEFT
      COMMON /GEAR0/ DTUSED,NQ,NSTEPS,NF,NJ
      DIMENSION U(2,31),XBKPT(31),SCTCH(10),WORK(5000),IWORK(500)

C      INITIALIZE PARAMETERS AND PDECOL CALLING ARGUMENTS
C
      NPDE = 2
      NINT = 30
      NPTS = NINT + 1
      KORD = 4
      NCC = 2
      TO = 0.0
      TOUT = 1.E-3
      DT = 1.E-7
      EPS = 1.E-4
      NF = 21
      INDEX = 1
      IWORK(1) = 5000
      IWORK(2) = 500
      DX = 1. / FLOAT(NPTS-1)
      DO 10 I=1,NPTS
      XBKPT(I) = FLOAT(I-1) * DX
10      CONTINUE
      XLEFT = XBKPT(1)

C      CALL THE PACKAGE TO INTEGRATE TO TIME T = TOUT
C
20      CALL PDECOL(TO,TOUT,DT,XBKPT,EPS,NINT,KORD,NCC,NPDE,NF,INDEX,
      .      WORK,IWORK)
C      CHECK FOR EXECUTION ERRORS
C
      IF ( INDEX .NE. 0 ) GO TO 70
C      OUTPUT PERFORMANCE DATA AND COMPUTED SOLUTION VALUES
C
      WRITE(3,30) TOUT,DTUSED,NSTEPS
30      FORMAT(//10X,3HT=,E10.3,7H DT=,E10.3,15H TOTAL STEPS=,I5)
      CALL VALUES(XBKPT,U,SCTCH,NPDE,NPTS,NPTS,0,WORK)
      DO 60 K=1,NPDE
60      K=1,NPDE
      WRITE(3,40) K
40      FORMAT(//10X,13HPDE COMPONENT,12/)
      WRITE(3,50) (U(K,I),I=1,NPTS)
50      FORMAT(10X,5E12.4)
60      CONTINUE

```

```

C SET N1 OUTPUT TIME AND CONTINUE THE INTEGRATION IF TOUT .LT. I
C OTHERS --, TERMINATE THE PROBLEM
C

```

```

      TOUT = TOUT + 10.
      IF ( TOUT .LT. 11. ) GO TO 20
      70 *SITE(3,80) INDEX = 13
      80 FORMAT(10X,7HINDEX= ,13)
      STOP
      END
      SUBROUTINE F(T,X,U,UX,FVAL,NPDE)
      DIMENSION U(NPDE), UX(NPDE), UX(NPDE), FVAL(NPDE)
      FVAL(1) = U(2)*U(2)*UX(1) - U(1)*U(2) - U(1)**2 + 10.
      * FVAL(2) = U(1)*U(2)*UX(1)
      * FVAL(2) = U(1)*U(1)*UX(2) + U(1)*U(2) - U(2)**2
      * FVAL(2) = U(1)*U(1)*UX(2) + 2.*U(1)*UX(1)*UX(2)
      RETURN
      END

```

```

      SUBROUTINE BNDY(T,X,U,UX,DBDU,DBDUX,DZDT,NPDE)
      DIMENSION U(NPDE), UX(NPDE), DZDT(NPDE)
      DIMENSION DBDU(NPDE,NPDE), DBDUX(NPDE,NPDE)
      COMMON /ENDPT/ KLEFT
      IF ( X .NE. KLEFT ) GO TO 10

```

```

      DBDU(1,1) = 1.
      DBDU(1,2) = 0.
      DBDU(2,1) = 0.
      DBDU(2,2) = 1.
      DBDUX(1,1) = 0.
      DBDUX(1,2) = 0.
      DBDUX(2,1) = 0.
      DBDUX(2,2) = 0.
      DZDT(1) = 0.
      DZDT(2) = 0.

```

```

      RETURN

```

```

10 DBDU(1,1) = U(2) * COS( U(1) * U(2) )
   DBDU(1,2) = U(1) * COS( U(1) * U(2) )
   DBDU(2,1) = U(2) * SIN( U(1) * U(2) )
   DBDU(2,2) = U(1) * SIN( U(1) * U(2) )
   DBDUX(1,1) = 1.
   DBDUX(1,2) = 0.
   DBDUX(2,1) = 0.
   DBDUX(2,2) = 1.
   DZDT(1) = 0.
   DZDT(2) = 0.

```

```

      RETURN

```

```

      END
      SUBROUTINE UINIT(X,U,NPDE)
      DIMENSION U(1)

```

```

C SET INITIAL CONDITIONS. NOTE THAT PI = 4.*ATAN(1.)
C

```

```

      U(1) = .5 * ( X + 1. )
      U(2) = 4.*ATAN(1.)

```

```

      RETURN

```

```

      END

```

```

      SUBROUTINE DERIV(T,X,U,UX,DFDU,DFDUX,DFDUX,NPDE)
      DIMENSION U(NPDE), UX(NPDE), UX(NPDE)
      DIMENSION DFDU(NPDE,NPDE), DFDUX(NPDE,NPDE), DFDUX(NPDE,NPDE)
      DFDU(1,1) = -U(2) - 2.*U(1)
      DFDU(1,2) = 2.*U(2)*UX(1) - U(1) + 2.*UX(2)*UX(1)
      DFDU(2,1) = 2.*U(1)*UX(2) + U(2) + 2.*UX(1)*UX(2)
      DFDU(2,2) = U(1) - 2.*U(2)

```

```

C

```

```

      DFDUX(1,1) = 2.*U(2)*UX(2)
      DFDUX(1,2) = 2.*U(2)*UX(1)
      DFDUX(2,1) = 2.*U(1)*UX(2)

```

```

C      ( U(2,2) = 2.*U(1)*U(1)
      DFUXX(1,1) = U(2)*U(2)
      DFUXX(1,2) = 0.
      DFUXX(2,1) = 1.
      DFUXX(2,2) = U(1)*U(1)
      RETURN
      END
C      PROGRAM SINE1,TSIN,TSTOUT,TAPE2=TSIN,TAPE3=TSTOUT)
C      THIS PROGRAM WAS WRITTEN TO BE RUN IN SINGLE PRECISION ON A CDC7600.
      DIMENSION XPLT(25),UPLT(2,2,404),SCRATCH(10)
      DIMENSION ERRAX(2),UEXACT(2),XBKPT(404),ID(80)
      DIMENSION KORD(30000),INCRK(1000)
      COMMON /GEAR/ MUDED,NQUSED,NS,NF,NJ
      COMMON /ISTAR/ I=1,IM2,IM3,TDUM(15)
      COMMON /PARAMS/ PI
C      READ IN AND INITIALIZE PDECOL CALLING ARGUMENTS AND PARAMETERS.
      PI = 4. * AT=N(1,0)
      READ(2,10) (ID(I),I=1,80)
      10 FORMAT(80A1)
      READ(2,20) NPDE,MF,TOUT
      20 FORMAT(2I3,E10.3)
      WRITE(3,10) (ID(I),I=1,80)
      WRITE(3,30) NPDE,MF,TOUT
      30 FORMAT(//8H NPDE = ,I2,5X,5HMF = ,I2,
      * 5X,26H RESULTS ARE FOR TIME T = ,E10.3//
      * 27H KORD NINT EPS ,24HINSTEPS NRES NJAC NO
      * 1X,30H ERROR(1)
      * 1X,30H ERROR(2) )
      40 READ(2,50) KORD,NINT,EPS
      50 FORMAT(2I3,E10.3)
      IF(KORD.LE.0) STOP
      NPTS = NINT + 1
      NCC = 2
      INCRK(1) = 30000
      INCRK(2) = 1000
      TO = 0.0
      DT = EPS * .5
      INDEX = 1
      NCPTS1 = KORD*NINT - NCC*(NINT-1) - 1
      DX = 1. / FLOAT(NINT)
      DO 60 I=1,NPTS
        XBKPT(I) = FLOAT(I-1)*DX
      60 CONTINUE
      CALL PDECOL(TO,TOUT,DT,XBKPT,EPS,NINT,KORD,NCC,NPDE,MF,INDEX,
      * WGRK,INCRK)
C      CHECK FOR ERRORS AND THEN COMPUTE AN ESTIMATE FOR THE OVERALL
C      MAXIMUM ERROR FOR EACH PDE COMPONENT. OUTPUT RESULTS.
      IF(INDEX.NE.0) WRITE(3,70) INDEX
      70 FORMAT(7H INDEX=,I3)
      DO 80 I=1,NPDE
        80 ERRMX(I) = 0.
      C      WORK(IW3) TO WORK(IW3+NCPTS1) CONTAINS THE COLLOCATION POINTS.
      DO 100 J=1,NCPTS1
        DX = (WORK(IW3+J)-WORK(IW3+J-1))/24.
        DO 90 I=2,24
          90 XPLT(I) = FLOAT(I-1)*DX + WORK(IW3+J-1)
          XPLT(1) = WORK(IW3+J-1)
          XPLT(25) = WORK(IW3+J)
          DO 100 I=1,25
            CALL VALUES(XPLT(I),UPLT(1,1),SCRATCH,1,1,0,WORK)
            CALL TRUSOL(XPLT(I),TOUT,UEXACT)
            DO 100 K=1,NPDE
              UPLT(K,2,1) = ABS( UPLT(K,1,I) - UEXACT(K) )
              IF(UPLT(K,2,1) .GT. ERRMX(K)) ERRMX(K) = UPLT(K,2,1)
            100 CONTINUE
          WRITE(3,110) KORD,NINT,EPS,NS,NF,NJ,NQUSED,(ERRMX(I),I=1,NPDE)

```

```

110 FQ=AT(14,16,1E14,3,1X,416,2E15,3)
CQ ) 40
END
C SUBROUTINES FOR SINE PROBLEM.
SUBROUTINE FIT(X,U,UX,UXX,FVAL,NPDE)
COMMON /PARAMS/ PI
FVAL = UXX + PI*PI*SIN(PI*X)
RETURN
END
SUBROUTINE BNDRY(T,X,U,UX,DBDU,DBDUX,DZDT,NPDE)
DBDU = 1.
DBDUX = 0.
DZDT = 0.
RETURN
END
SUBROUTINE UINIT(X,U,NPDE)
DIMENSION U(NPDE)
CALL TRUSOL(X,0.,U)
RETURN
END
SUBROUTINE DERIV(T,X,U,UX,UXX,DFDU,DFDUX,DFDUXX,NPDE)
DFDUXX = 1.
DFDUX = 0.
DFDU = 0.
RETURN
END
SUBROUTINE TRUSOL(X,T,U,UXACT)
COMMON /PARAMS/ PI
UXACT = 1. + SIN(PI*X) * ( 1. - EXP(-PI*PI*T) )
RETURN
END
SINE PROBLEM INPUT DATA
1 21 .1
3 1 .01
3 3 .001
3 10 .00003
3 20 .00001
3 60 .000001
3 100 .000001
4 1 .01
4 2 .001
4 4 .00001
4 8 .000001
4 16 .0000001
5 1 .0001
5 2 .00001
5 4 .000001
5 8 .00000001
6 1 .0001
6 2 .000001
6 4 .00000001
-1
PROGRAM TWO(TSIN,ISTOUT,TAPE2=TSIN,TAPE3=ISTOUT)
C THIS PROGRAM WAS WRITTEN TO BE RUN IN SINGLE PRECISION ON A CDC7600.
DIMENSION APLT(25),UPLT(2,2,404),SCRICH(10)
DIMENSION ERRMX(2),UEXACT(2),X8KPY(404),ID(80)
DIMENSION WORK(30000),IMORK(1000)
COMMON /GEAR/ HUSED,NOUSED,NS,NF,NJ
COMMON /ISTART/ IM1,IM2,IM3,IDUM(15)
COMMON /PARAMS/ PI
C READ IN AND INITIALIZE PDECOL CALLING ARGUMENTS AND PARAMETERS.
PI = 4. * ATAN(1.0)
READ(2,10) (ID(I),I=1,80)
10 FORMAT(80A1)
READ(2,20) NPDE,MF,TOUT
20 FORMAT(2I3,E10,3)

```



```

      DFDUX(1,1) = 0.
      DFDUX(1,2) = 0.
      DFDUX(2,1) = 0.
      DFDUX(2,2) = 0.
      RETURN
10 CONTINUE
      DZDT(1) = 0.
      DZDT(2) = 0.
      DEDU(1,1) = 3.
      DEDU(1,2) = 0.
      DEDU(2,1) = -5.4598150033142E+01
      DEDU(2,2) = 0.
      DEDU(1,1) = 1.
      DEDU(1,2) = 0.
      DEDU(2,1) = 0.
      DEDU(2,2) = 5.
      RETURN
      END
      SUBROUTINE UINIT(X,U,NPDE)
      DIMENSION U(NPDE)
      CALL TRUSOL(X,0.,U)
      RETURN
      END
      SUBROUTINE DERIV(T,X,U,UX,UXX,DFDU,DFDUX,DFDUXX,NPDE)
      DIMENSION U(NPDE),UX(NPDE),UXX(NPDE),DFDU(NPDE),DFDUX(NPDE),DFDUXX(NPDE)
      DFDU(1,1) = 16.*X*T - 2.*T - 16.*(U(2)-1.0)
      DFDU(1,2) = UX(1) - 16.*(U(1)-1.0)
      DFDU(2,1) = 4.
      DFDU(2,2) = 0.
      DFDUX(1,1) = UX(2)
      DFDUX(1,2) = UX(1)
      DFDUX(2,1) = 1.
      DFDUX(2,2) = 0.
      DFDUXX(1,1) = U(2) - 1.0
      DFDUXX(1,2) = 0.
      DFDUXX(2,1) = 0.
      DFDUXX(2,2) = 1.
      RETURN
      END
      SUBROUTINE TRUSOL(X,T,UEXACT)
      DIMENSION UEXACT(2)
      UEXACT(1) = 10.*X*T*EXP(-4.*X) + 1.0
      UEXACT(2) = 1.*X*X + 1.0
      RETURN
      END
      TWO PDE PROBLEM INPUT DATA
      2 21 1.0
      3 3 .0027
      3 6 .00064
      3 12 .00016
      3 24 .00004
      3 48 .00001
      3 96 .0000025
      4 1 .015
      4 2 .0013
      4 4 .0007
      4 8 .00004
      4 16 .0000025
      4 32 .000000015
      4 64 .000000001
      5 1 .0035
      5 2 .0001
      5 4 .000003
      5 8 .00000006
      5 16 .000000001

```



```

6 1 .0046
6 2 .013
6 4 .0000013
6 8 .00000001
-1
C THIS PROGRAM WAS WRITTEN TO BE RUN IN SINGLE PRECISION ON A CDC7600.
C DIMENSION XPLT(25),UPLT(2,2,404),SCRATCH(10)
C DIMENSION ERRMX(2),UEXACT(2),XBKPT(404),ID(80)
C DIMENSION WORK(30000),IWORK(1000)
C COMMON /GEAR/ MUUSED,NOUSED,NS,NF,NJ
C COMMON /ISTART/ IM1,IM2,IM3,IDUM(15)
C COMMON /PARAMS/ PI
C READ IN AND INITIALIZE PDECOL CALLING ARGUMENTS AND PARAMETERS.
PI = 4. * ATAN(1.0)
READ(2,10) (ID(I),I=1,80)
10 FORMAT(80A1)
READ(2,20) NPDE,MF,TOUT
20 FORMAT(2I3,E10.3)
WRITE(3,10) (ID(I),I=1,80)
WRITE(3,30) NPDE,MF,TOUT
30 FORMAT(/,8X, NPDE = ,12.5X,5HMF = ,12.
* 5X,26H RESULTS ARE FOR TIME T = ,E10.3//
* 24HNSTEPS NRES NJAC NQ
* 27H KORD NINT EPS ERROR(2) )
* 1X,30H ERROR(1)
40 READ(2,50) KORD,NINT,EPS
50 FORMAT(2I3,E10.3)
IF(KORD.LE.0) STOP
NPTS = NINT + 1
NCC = 2
IWORK(1) = 30000
IWORK(2) = 1000
T0 = 0.0
DT = EPS * .5
INDEX = 1
DT = EPS * .5
INDEX = 1
NCPST1 = KORD*NINT - NCC*(NINT-1) - 1
DX = 1. / FLOAT(NINT)
DO 60 I=1,NPTS
XBKPT(I) = FLOAT(I-1)*DX
60 CONTINUE
CALL PDECOL(T0,TOUT,DT,XBKPT,EPS,NINT,KORD,NCC,NPDE,MF,INDEX,
* WORK,IWORK)
C CHECK FOR ERRORS AND THEN COMPUTE AN ESTIMATE FOR THE OVERALL
C MAXIMUM ERROR FOR EACH PDE COMPONENT. OUTPUT RESULTS.
IF(INDEX.NE.0) WRITE(3,70) INDEX
70 FORMAT(7H INDEX=,I3)
DO 80 I=1,NPDE
80 ERRMX(I) = 0.
C WORK(IW3) TO WORK(IW3+NCPST1) CONTAINS THE COLLOCATION POINTS.
DO 100 J=1,NCPST1
DX = (WORK(IW3+J)-WORK(IW3+J-1))/24.
DO 90 I=2,24
90 XPLT(I) = FLOAT(I-1)*DX + WORK(IW3+J-1)
XPLT(1) = WORK(IW3+J-1)
XPLT(25) = WORK(IW3+J)
DO 100 I=1,25
CALL VALUES(XPLT(I),UPLT(1,1,I),SCRATCH,1,1,1,0,WORK)
CALL TRUSOL(XPLT(I),TOUT,UEXACT)
DO 100 K=1,NPDE
UPLT(K,2,1) = ABS( UPLT(K,1,1) - UEXACT(K) )
IF(UPLT(K,2,1) .GT. ERRMX(K)) ERRMX(K) = UPLT(K,2,1)
100 CONTINUE
WRITE(3,110) KORD,NINT,EPS,NS,NF,NJ,NOUSED,(ERRMX(I),I=1,NPDE)
110 FORMAT(14,I6,1E14,3,1X,4I6,2E15,3)

```

```

GO TO 40
EX
C SUBROUTINE FOR BURGERS EQUATIONS PROBLEM.
SUBROUTINE FIT(X,U,UX,UXX,FVAL,NPDE)
FVAL = .003*UXX - U*UX
RETURN
END
SUBROUTINE BNDRY(T,X,U,UX,DBOU,DBDUX,DZOT,NPDE)
IF( X.NE. 0. ) GO TO 10
DZOT = 0.
DBOU = 1.
DBDUX = 0.
RETURN
10 CONTINUE
DZOT = 0.
DBOU = 0.
DBDUX = 0.
RETURN
END
SUBROUTINE UNIT(X,U,NPDE)
DIMENSION U(NPDE)
CALL TRUSOL(X,O.,U)
RETURN
END
SUBROUTINE DERIV(T,X,U,UX,DFDU,DFDUX,DFDUXX,NPDE)
DFDUXX = .003
DFDUX = -U
DFDU = -UX
RETURN
END
SUBROUTINE TRUSOL(X,T,UEXACT)
ANUIN = 1./ .006
A = -(X-.5 + 4.95*T)*ANUIN*.1
B = -(X-.5 + 0.75*T)*ANUIN*.5
C = -(X-.375) * ANUIN
A = EXP(A)
B = EXP(B)
C = EXP(C)
UEXACT = (.1*A + .5*B + C) / (A + B + C)
RETURN
END
BURGERS EQUATION PROBLEM INPUT DATA
1 21 .8
3 50 .003
3100 .0003
3200 .0001
3400 .00004
4 20 .004
4 40 .001
4 80 .0001
4160 .00001
5 15 .004
5 30 .001
5 60 .00003
5100 .000003
6 10 .01
6 20 .001
6 40 .00003
6 80 .000001
-1
C PROGRAM WAVE(TSIN,TSTOUT,TAPE2=TSIN,TAPE3=TSTOUT)
C THIS PROGRAM WAS WRITTEN TO BE RUN IN SINGLE PRECISION ON A CDC7600.
DIMENSION XPLI(25),UPLI(2,2,404),SCRCH(10)
DIMENSION ERMA(2),UEXACT(2),XKPT(404),ID(80)
DIMENSION WORK(30000),IWORK(1000)
COMMON /GEAR0/ HUSED,NQUSED,NS,NF,NJ

```

```

COMMON /ISTART/ IW1,IW2,IW3,IOWM(15)
C ( JH /PARAMS/ PI
C READ I... AND INITIALIZE PDECOL CALLING ARGUMENTS AND PARAMETERS.
PI = 4. * ATAN(1.0)
READ(2,10) (ID(I),I=1,80)
10 FORMAT(80A1)
READ(2,20) NPDE,MF,TOUT
20 FORMAT(2I3,E10.3)
WRITE(3,10) (ID(I),I=1,80)
WRITE(3,30) NPDE,MF,TOUT
30 FORMAT(//8H NPDE = 12.5X,5HMF = 12.5X,5HTOUT = 12.5X,5H
* 27H KORD NINT EPS .24HINSTEPS NRES NJAC NO .
* 1X,30H ERROR(1)
40 READ(2,50) KORD,NINT,EPS
50 FORMAT(2I3,E10.3)
IF(KORD.LE.0) STOP
NPTS = NINT + 1
NCC = 2
IWORK(1) = 30000
IWORK(2) = 1000
T0 = 0.0
DT = EPS * .5
INDEX = 1
NCPTS1 = KORD*NINT - NCC*(NINT-1) - 1
DX = 1. / FLOAT(NINT)
DO 60 I=1,NPTS
XBKPT(I) = FLOAT(I-1)*DX
60 CONTINUE
CALL PDECOL(T0,TOUT,DT,XBKPT,EPS,NINT,KORD,NCC,NPDE,MF,INDEX,
* WORK,IWORK)
C CHECK FOR ERRORS AND THEN COMPUTE AN ESTIMATE FOR THE OVERALL
C MAXIMUM ERROR FOR EACH PDE COMPONENT. OUTPUT RESULTS.
IF(INDEX.NE.0) WRITE(3,70) INDEX
70 FORMAT(7H INDEX=.13)
DO 80 I=1,NPDE
80 ERRMX(I) = 0.
C WORK(IW3) TO WORK(IW3+NCPTS1) CONTAINS THE COLLOCATION POINTS.
DO 100 J=1,NCPTS1
DX = (WORK(IW3+J)-WORK(IW3+J-1))/24.
DO 90 I=2,24
90 XPLT(I) = FLOAT(I-1)*DX + WORK(IW3+J-1)
XPLT(1) = WORK(IW3+J-1)
XPLT(25) = WORK(IW3+J)
DO 100 I=1,25
CALL VALUES(XPLT(I),UPLT(1,1,1),SCRATCH,1,1,1,0,WORK)
CALL TRUSOL(XPLT(I),TOUT,UEXACT)
DO 100 K=1,NPDE
UPLT(K,2,1) = ABS( UPLT(K,1,1) - UEXACT(K) )
IF(UPLT(K,2,1) .GT. ERRMX(K)) ERRMX(K) = UPLT(K,2,1)
100 CONTINUE
WRITE(3,110) KORD,NINT,EPS,NS,NF,NJ,NQ,USED,(ERRMX(I),I=1,NPDE)
110 FORMAT(14,I6,1E14,3,I8,4I6,2E15,3)
GO TO 40
END
C SUBROUTINES FOR WAVE PROPAGATION PROBLEM.
SUBROUTINE F(T,X,U,UX,FVAL,NPDE)
FVAL = -UX
RETURN
END
SUBROUTINE BNDRY(T,X,U,UX,DBDU,DBDUX,DZDT,NPDE)
COMMON /PARAMS/ PI
IF( X.NE. 0. ) GO TO 10
DZDT = -2.*PI*COS(-4.*PI*T)
DBDU = 1.
DBDUX = 0.

```

```

      10 CALL UINIT
      DZDT = 0.
      DBDU = 0.
      DBDUX = 0.
      RETURN
    END
    SUBROUTINE UINIT(X,U,NPDE)
      DIMENSION U(NPDE)
      CALL TRUSOL(X,0.,U)
      RETURN
    END
    SUBROUTINE DERIVE(T,X,U,UX,UXX,DFDU,DFDUX,DFDUXX,NPDE)
      DFDU = 0.
      DFDUX = -1.
      DFDUXX = 0.
      RETURN
    END
    SUBROUTINE TRUSOL(X,T,U,UEXACT)
      COMMON /PARAMS/ PI
      UEXACT = 1.0 + .5*SIN(4.*PI*(X-T))
      RETURN
    END
    WAVE PROPAGATION PROBLEM INPUT DATA
    1 11 .5
    3 20 .001
    3 40 .001
    3 80 .00001
    3 160 .000001
    3 320 .0000001
    4 5 .001
    4 10 .0001
    4 20 .00001
    4 40 .000001
    4 80 .00000001
    4 1601 .0000E-10
    5 3 .001
    5 6 .0004
    5 12 .000002
    5 24 .00000001
    5 481 .0000E-10
    6 2 .001
    6 4 .0001
    6 8 .0000001
    6 16 .00000001
    6 321 .0000E-10

```

279

```

IF NO.LE.O.CR.MAP.LE.O.OR.MAR.LE.2)GO TO 610
IF NO.GE.MAP.CR.MAR.GE.2)GO TO 610
IF MAP.GT.NOPTS.CR.MAP.GE.100)GO TO 610
IF NOPTS.LE.O.OR.NOVS.LE.O.OR.NO.GT.610)GO TO 610
IF NO.LE.O.OR.NOVS.GT.NO)GO TO 610
IF NO.GT.NOVS)GO TO 610
GO TO 615
610 CONTINUE
WRITE(KFILE,611)
611 FORMAT(//,' $$$ PROBLEMS WITH ARGUMENT LIST. PLOT NOT PERFORMED.',
1)
ARTEINFILE,612)MAD,MAP,MAR,NOPTS,NOVS,NO
612 FORMAT(1X,618)
RETURN
C
500 NSUB = AMAX + EPS
MEXP1 = MEXP
SUBT = NSUB
DO 520 N = MAD,MAP,MAR
DO 520 L = NO,NOVS
520 Y(N,L) = Y(N,L) - SUBT
615 CONTINUE
AMAX = -1.0E+20
AMIN = 1.0E+20
DO 620 N = MAD,MAP,MAR
DO 620 L = NO,NOVS
IF(Y(N,L).GT.AMAX)AMAX = Y(N,L)
IF(Y(N,L).LT.AMIN)AMIN = Y(N,L)
620 CONTINUE
IPT = 0
JPT = 0
FAC = ABS(AMAX-AMIN)
IF(FAC.GT.1.0E-16)GO TO 630
WRITE(KFILE,623)
629 FORMAT(//,' $$$ (AMAX - AMIN) IS LESS THAN 1.0E-16, PLOT NOT',
1, PERFORMED.,/)
RETURN
C
C AMIN---AMAX IS THE RANGE FROM THE SMALLEST TO THE LARGEST VALUE
C PLOTTED. IF AMIN AND AMAX ARE BOTH POSITIVE OR BOTH NEGATIVE, THE
C PROCEDURE FOR SELECTING THE SCALE IS CONTAINED IN STATEMENTS 630
C TO 702. IF THE RANGE INCLUDES ZERO, HOWEVER, THE SCALE IS FOUND
C IN STATEMENTS 800 TO 860.
C
630 DMAX = AMAX
DMIN = AMIN
FAC = AMAX*AMIN
IF(FAC.LT.ZERO)GO TO 800
FAC = ONE
IF(AMAX.GE.ZERO.AND.AMIN.GE.ZERO)GO TO 632
DMAX = -AMIN
DMIN = -AMAX
FAC = -ONE
C
C STATEMENTS 632-663 SCALE DOWN (OR UP) AMAX, AMIN, AND THE DATA (BY
C THE FACTOR TEN**MEXP) SO THAT AMAX FALLS IN THE RANGE 1 TO 19.....
C
632 IF(DMAX-ONE)635,660,640
635 DMAX = DMAX*1.0E+03
IPT = IPT + 1
GO TO 632
640 IF(DMAX-19.E+00)660,660,661
650 DMAX = DMAX/TEN
JPT = JPT + 1
GO TO 640
660 MEXP = 3*IPT - JPT

```

```

IF MEXP.EQ.0)GO TO 663
F( TEN**-(MEXP)
  AMAX=AMAX*FX
  AMIN=AMIN*FX
DO 661 N = MAD,MAP,VAR
DO 661 L = NO,NOVS
  YIN(L) = YIN(L)*FX
661 CONTINUE
663 CONTINUE
  IF FAC.LT.ZERO)GO TO 652
  DMAX = AMAX
  DMIN = AMIN
  GO TO 655
652 DMAX = -AMIN
  DMIN = -DMAX
655 CONTINUE
C
C STEPS UP TO 670 DECIDE 'M', THE NUMBER OF DIGITS PAST THE DECIMAL
C POINT IN ALL THE SCALE POINTS. LATER, MT = 3 + M, WHERE MT IS
C THE NUMBER OF DIGITS IN THE SCALE POINT, COUNTING THE DECIMAL
C AND 2 DIGITS TO ITS LEFT.
C
  IF (SUB.NE.0)FACN = FACN*TEN**-(MEXP)
  M = 0
  MT = 3
  NMAX = DMAX + EPS*FACN
  NMIN = DMIN - EPS*FACN
  ENOP = NMIN
  IF (DMIN.EQ.ZERO)NMIN = -1
  INDEX(1,3) = NMIN
  NMIN = NMIN + 1
  NMAX = NMAX - NMIN
  IF (NRAN-1)665,670,670
662 NMIN = NMIN + 1
  NMAX = NMAX - NMIN
  IF (NRAN-1)665,670,670
665 M = M + 1
  DMIN = (DMIN-ENOP)*TEN - EPS*FACN
  DMAX = (DMAX-ENOP)*TEN + EPS*FACN
  NMAX = DMAX
  NMIN = DMIN
  MT = 3 + M
  IF (MT.GT.8)GO TO 500
  INDEX(1,MT) = NMIN
  ENOP = NMIN
  GO TO 662
670 INDEX(1,MT) = INDEX(1,MT) + 1
C
C STEPS UP TO 680 FIX THE LAST DIGITS OF THE SCALE POINTS---THAT IS,
C FIX INDEX(L,M) FOR ALL L, FOR THE LARGEST M USED (MT). NRAN IS THE
C NUMBER OF SCALE POINTS TO BE USED.....
C
  NRAN = NRAN + 1
  IF (MT.EQ.7).AND.(NRAN.GT.17))GO TO 500
  IF (MT.CO.8).AND.(NRAN.GT.15))GO TO 500
  IPT = INDEX(1,MT)
  DO 680 MTA = 2,NRAN
  IPT = IPT + 1
  IPT = INDEX(MTA,MT) = IPT
680
C
C STEPS 685-690 CORRECT THE LAST DIGIT OF THE SCALE POINTS, IF NE-
C CCESSARY, AND DETERMINE ALL OTHER DIGITS TO THE RIGHT OF THE DE-
C CIMAL---THAT IS, FIX INDEX(L,M) FOR ALL L AND ALL M>3.....
C
  IF (M-1)693,685,685
685 DO 690 LT = 1,M
  NPT = MT - LT
  DO 690 MTA = 1,NRAN

```



```

      MEXP = 21 - N
      DX = TEN**(MEXP)
      IF (AMAX.GT.DMIN)GO TO 810
801 CONTINUE
      RETURN
810 DIFF = DMIN
      DMIN = ABS(AMIN)
      DMAX = ABS(AMAX)
      NRAN = 1
      DO 815 N = 1,40
        PN = N
        DUFF = PN*DIFF
        IF(DUFF.GT.DMIN.AND.DUFF.GT.DMAX)GO TO 816
        IF(DUFF.LE.DMAX)NRAN = NRAN + 1
        IF(DUFF.LE.DMIN)NRAN = NRAN + 1
        EMOP = DUFF
815 CONTINUE
      RETURN
816 IF(NRAN.GE.41)GO TO 820
      JPT = 1
      B4C = DIFF/TEN
      DO 817 N = 1,40
        PN = N
        DUFF = PN*B4C
        IF(DUFF.GT.DMAX.AND.DUFF.GT.DMIN)GO TO 818
        IF(DUFF.LE.DMAX)JPT = JPT + 1
        IF(DUFF.LE.DMIN)JPT = JPT + 1
        ETA1 = DUFF
817 CONTINUE
      818 IF((NRAN.EQ.2.AND.JPT.GT.19).OR.(NRAN.EQ.3.AND.JPT.GT.15))GO
        1 TO 820
        DIFF = B4C
        NRAN = JPT
        EMOP = ETA1
        MEXP = MEXP - 1
820 CONTINUE
      M = 1
      IF(EMOP.LT.99.0E+00.AND.EMOP.GT..99E-02)GO TO 830
      FX = TEN**(MEXP)
      DIFF = DIFF*FX
      AMAX = AMAX*FX
      AMIN = AMIN*FX
      DO 825 N = MAD,MAP,MAR
        DO 825 L = NO,NOVS
          YIN(L) = YIN(L)*FX
825 CONTINUE
      GO TO 831
830 IF(EMEXP.LT.0)M = -MEXP
      MEXP = 0
831 CONTINUE
C
C STEPS THROUGH 860 FIX MT, INDEX, AND XSCALE. THE FIRST THREE DI-
C GITS OF INDEX ARE FIXED BEFORE STATEMENT 855.....
C
      EMOP = ZERO
      DMIN = ABS(AMIN)
      MTA = 0
839 IF(EMOP.GT.DMIN)GO TO 840
      DMAX = EMOP
      EMOP = EMOP + DIFF
      MTA = MTA + 1
      IF(MTA.GT.100)RETURN
      GO TO 839
840 XSCALE(1) = -DMAX
      DO 842 N = 2,NRAN

```


AD-A139 604

MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND
ELECTRON TRANSPORT VOLUME 2(U) ARCON CORP WALTHAM MA
S WOOLF ET AL. JAN 84 RADC-TR-83-313-VOL-2

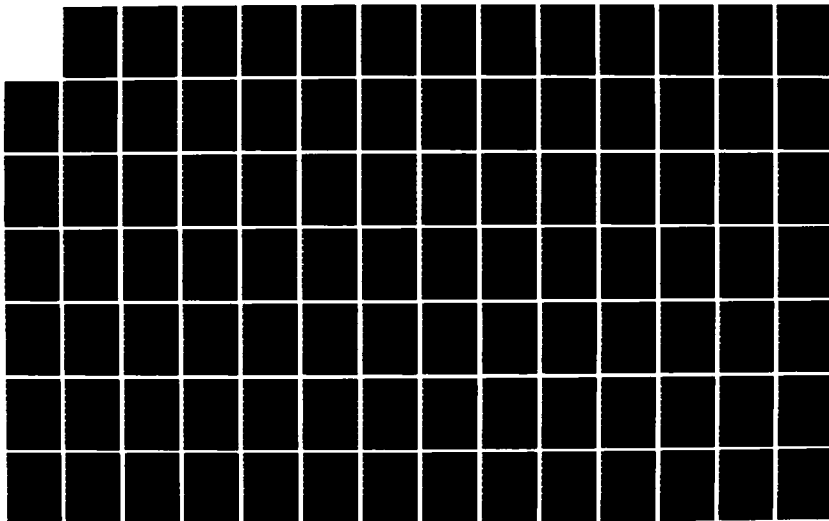
4/5

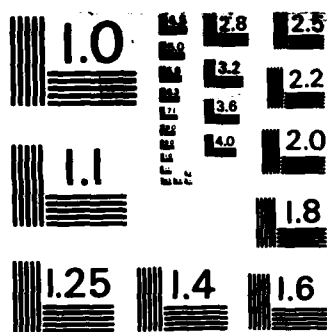
UNCLASSIFIED

F19628-78-C-0188

F/G 9/2

NL





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

```

708 IF (N1-31711.710,711
709 J1 = N1) = 1H0
711 M1 = M1A + 1
      IF (M1A.GT.NRAN)GO TO 715
      ENOP = XSCALE(M1A)
      IF (ENOP.EQ.ZERO)NMAX = 1H
      IF (ENOP.GT.ZERO)NMAX = 1H+
      ETAI = (SMOP - AMIN)*BAC + 6.0E+00 + EPS*FACN
      JPT = ETAI
712 CONTINUE
715 NMAX = - MEXP - MEXP1
      IF (NMAX)713,714,713
713 WRITE(KFILE,719)NMAX
719 FORMAT(49X,'MULTIPLY SCALE BY 10 TO THE',13)
714 CONTINUE
      .....
      STATEMENT 720 PRINTS THE SCALE
      .....
      STATEMENTS AFTER 720 FIND THE N FOR WHICH IETA(N)(THE VARIABLE CO-
      DED FOR THE SYMBOLS TO BE PRINTED) SHOULD BE CHANGED FROM BLANK TO
      THE APPROPRIATE SYMBOL AND THEN PRINT IETA(N) FOR ALL N ACROSS THE
      PAGE.
      .....
      WRITE(KFILE,720)(IETA(N),N=3,128)
720 FORMAT(2X,50A1,50A1,26A1)
      DO 730 L = MAD,MAP,MAR
      DO 725 N = 1,128
725 IETA(N) = 1H
      WRITE(KFILE,738)L,L
      DO 730 M = NO,NOVS
      ETAI = (Y(L,M)-AMIN)*BAC + 6.0E+00 + EPS*FACN
      NT = ETAI
738 FORMAT(14,61X,58X,14)
      GO TO (739,740,741,742,743,744),M
739 IETA(NT) = 1H+
      GO TO 735
740 IETA(NT) = 1H0
      GO TO 735
741 IETA(NT) = 1HX
      GO TO 735
742 IETA(NT) = 1H+
      GO TO 735
743 IETA(NT) = 1H=
      GO TO 735
744 IETA(NT) = 1H$
735 WRITE(KFILE,732)(IETA(N),N=6,123)
730 CONTINUE
732 FORMAT(1H+,4X,50A1,50A1,18A1)
      WRITE(KFILE,720)(IETA(N),N=3,128)
      IF (NMAX)733,737,733
733 WRITE(KFILE,719)NMAX
737 CONTINUE
      IF (MEXP.EQ.0)GO TO 736
      FX = TEN**(-MEXP)
      DO 734 N = MAD,MAP,MAR
      DO 734 L = NO,NOVS
      Y(N,L) = Y(N,L)*FX
734 CONTINUE
736 CONTINUE
      IF (NSUB.EQ.0)GO TO 750
      ENOP = SUBT*TEN**(-MEXP1)
      DO 745 N = MAD,MAP,MAR

```

```

D( 45 L = NC.NOVS
745 Y(...L) = Y(N,L) + SUBT
WRITE(KFILE,746)EMOP
746 FORMAT(18X,*IT WAS NECESSARY TO SUBTRACT*.F8.1,* FROM THE*,
1* VARIABLES PLOTTED TO MAKE THE SCALE MEANINGFUL*)
WRITE(KFILE,747)
747 FORMAT(41X,* VALUES OF Y(N,L) RETURNED ARE ORIGINAL VALUES*)
750 CONTINUE
IF(MEXP1.EQ.0)GO TO 760
FX = TEN*(-MEXP1)
DO 755 N = MAD,MAR,MAR
DO 755 L = NO.NOVS
755 Y(N,L) = Y(N,L)*FX
760 RETURN
END

```

Program Listing

- ONEDSN -

(Reference: Vol. I, Section IV.A.2)

```

C-----
C PROGRAM ONEDSN(INPUT,OUTPUT)
C-----
C ONEDSN SOLVES THE ONE-DIMENSIONAL STEADY STATE DISCRETE ORDINATE
C FORM OF THE TRANSPORT EQUATION IN SLAB GEOMETRY FOR THE CASES OF
C ISOTROPIC AND SCREENED-RUTHERFORD SCATTERING WITH SURFACE OR
C DISTRIBUTED PARTICLE SOURCES. OTHER SCATTERING MODELS CAN BE
C INCORPORATED BY SUBSTITUTING AN APPROPRIATE EXPRESSION FOR THE
C ANGULAR REDISTRIBUTION FUNCTION ARRAY "SCAT(K,L,M)" IN SUBROUTINE
C "INPUTS". THE NUMBER OF DISCRETE ORDINATES CAN BE VARIED UP TO A
C MAXIMUM OF 12 FORWARD DIRECTIONS (24 DISCRETE ORDINATES IN ALL).
C UP TO 100 CELLS ARE ALLOWED FOR THE SPATIAL DISCRETIZATION. THE
C DIAMOND DIFFERENCE ITERATION SCHEME IS USED. NO CONVERGENCE
C ACCELERATION SCHEME IS EMPLOYED, SO THAT EACH ITERATION
C CORRESPONDS TO A PHYSICAL ORDER OF SCATTERING.
C-----
C COMMON/A/SCAT(24,24),AMU(24),SURF(24),A(12,2),B(12,2),EFLUX(101),
C +,CUR(24),S(100,24),SDISTR(100,24),CFLUX(100,24),EF(100),X(101),
C +WGT(24),ERR,ZERO,ETA(2),NX,NANG,NPOS,NSCAT,NORDS,MT,
C +KSRC,SIG(2),ALBEDO(2),NINT,DX
C-----
C ITMAX      = MAXIMUM NO. OF ITERATIONS
C NORDS      = 1 IF PRINT OUT OF ORDERS-OF-SCATTERING FLUX IS
C              DESIRED. NO PRINTOUT FOR OTHER VALUES OF N.
C NSCAT      = 0 IF ISOTROPIC SCATTERING IS ASSUMED
C              = 1 IF SCREENED-RUTHERFORD OR OTHER NON-ISOTROPIC
C              SCATTERING LAW IS ASSUMED
C DX         = WIDTH OF X-CELL
C SIG        = 1/COLLISION MEAN FREE PATH
C ALBEDO     = PROBABILITY OF A PARTICLE BEING SCATTERED (NOT
C              ABSORBED).
C ERR        = FRACTIONAL ERROR IN THE SOURCE FUNCTION AT WHICH
C              ITERATIONS ARE TERMINATED.
C ZERO       = LEVEL AT WHICH ERRORS IN THE CALCULATED SOURCE ARE
C              IGNORED
C NX         = NO. OF X-CELLS
C NPOS       = NO. OF POSITIVE DIRECTIONS (=NANG/2)
C ETA        = SCREENING PARAMETER USED WHEN SCREENED-
C              RUTHERFORD SCATTERING IS ASSUMED (NSCAT=1);
C              ALSO ANISOTROPY PARAMETER WHEN NON-ISOTROPIC
C              SCATTERING OTHER THAN SCREENED-RUTHERFORD IS
C              ASSUMED.
C KSRC       = 0 IF INITIAL CONDITION IS SURFACE SOURCE ONLY
C              = 1 IF NON-ZERO INITIAL DISTRIBUTED SOURCE ONLY
C              = 2 IF BOTH SURFACE AND DISTRIBUTED SOURCES ARE
C              PRESENT INITIALLY
C NCASE      = PARAMETER TO SELECT FORM OF SURFACE SOURCE
C              ANGULAR DEPENDENCE
C              = 1, COSINE SOURCE
C              = 2, (COSINE)**2 SOURCE
C              = 3, (COSINE)**3 SOURCE
C NINT       = INDEX OF FIRST X-CELL BEYOND MATERIAL INTERFACE.
C              IF NO MATERIAL INTERFACE PRESENT, READ IN NINT = 0.
C-----
C-----DERIVED PARAMETERS AND BOUNDARY CONDITIONS
C SCAT(K,L,M) = SCATTERING KERNEL
C              (K = INDEX OF INITIAL DIRECTION DISCRETE ORDINATE
C              L = INDEX OF FINAL DIRECTION DISCRETE ORDINATE
C              M = INDEX OF MATERIAL LAYER)
C NANG       = TOTAL NO. OF DISCRETE ORDINATES (ANGLES)

```



```

C      Y 'J)      * DIRECTION COSINES OF POSITIVE DIRECTIONS
C      (K)      * INCIDENT ANGULAR FLUX AT X = 0.
C      ( X IS THE SPATIAL COORDINATE; X = 0 IS THE LEFT
C      BOUNDARY)
C
C      SDISTR(I,K) = INITIAL DISTRIBUTED SOURCE AT CENTER OF ITH CELL
C      IN KTH DIRECTION
C-----CALCULATED QUANTITIES-----
C      CFUX(I,K) = ANGULAR FLUX AT CENTER OF ITH CELL IN KTH DIRECTION
C      EF(I,K) = ANGULAR FLUX AT CELL EDGE
C      EFUX(I) = SCALAR FLUX AT CELL EDGE
C      CUR(K) = PARTICLE CURRENT IN KTH DIRECTION
C      S(I,K) = SCATTERED SOURCE FUNCTION
C      QUT = TOTAL EMERGENT CURRENT FROM SLAB
C-----READ INPUT PARAMETERS-----
C      READ 555,ITMAX,NORDS,NSCAT,KSRC
C
C      SUBROUTINE 'INPUTS'
C      1) READS IN REMAINING INPUT PARAMETERS
C      2) INITIALIZES GAUSS QUADRATURE ANGLES AND WEIGHTS
C      3) COMPUTES THE SCATTERING PROBABILITY FUNCTION
C      4) INITIALIZES THE SOURCE FUNCTION
C
C      CALL INPUTS
C-----PRINT OUT DERIVED CONSTANTS, BOUNDARY CONDITIONS, ETC.-----
C      PRINT 502
C      *AMU* IS ARRAY OF GAUSS QUADRATURE COSINES
C      PRINT 500,(AMU(J),J=1,NPOS)
C      PRINT 505
C      *SURF* IS ARRAY OF SURFACE SOURCE VALUES AT DISCRETE ORDINATES
C      PRINT 500,(SURF(J),J=1,NPOS)
C
C      *SUM* IS SOURCE NORMALIZATION FACTOR = INTEGRAL OF SOURCE OVER
C      ANGLE
C      SUM=0.
C      DO 100 J=1,NPOS
C      SUM=SUM+SURF(J)*WGT(J)
C      *A,B* ARE CONSTANTS THAT ARE USED REPETITIVELY IN THE ITERATIONS
C      DO 100 M=1,INT
C      A(J,M)=2.*AMU(J)/(2.*AMU(J)+SIG(M)*DX)
C      B(J,M)=DX/(2.*AMU(J)+SIG(M)*DX)
C      PRINT 514,SUM
C-----CONVERT FROM CURRENT SOURCE TO FLUX SOURCE-----
C      DO 10 J=1,NPOS
C      10 SURF(J)=SURF(J)/AMU(J)
C-----ZERO OUT THE FLUX-----
C      DO 11 K=1,NANG
C      DO 11 I=1,NX
C      CFUX(I,K)=0.
C      NXPN=NX+1
C
C      DO 12 I=1,NXP
C      X(I)=(I-1)*DX
C      NANG=2*NPOS
C      NANGP=NANG+1
C-----MAIN ITERATION LOOP-----
C      DO 350 IT=1,ITMAX
C      OUT=TRAN=BACK=0.
C
C      PRINT OUT ORDERS OF SCATTERING SCALAR FLUX VALUES IF DESIRED
C      IF(NORDS.EQ.1)CALL INTPRINT

```

```

C      ( - FLUXES(TRAN,1,0,1,0)
C      CALL FLUXES(BACK,-1,NANGP,-1,NXP)
C      OUT=TRAN+BACK
C-----CALCULATE NEW SOURCE-----
C      EMAX=0.
C      M=1
C      DO 200 I=1,NX
C      IF(I.GE.NINT)M=2
C      DO 200 K=1,NANG
C      SUM=SDISTR(I,K)
C      DO 150 L=1,NANG
C      150 SUM=SUM+SCAT(K,L,M)*CFUX(I,L)*NGT(L)*ALBEDO(M)*SIG(M)
C      IF(ABS(SUM).LT.ZERO)GO TO 200
C      ERROR=ABS((S(I,K)-SUM)/SUM)
C      IF(ERROR.GT.EMAX)EMAX=ERROR
C      200 S(I,K)=SUM
C      IF(EMAX.LT.ERR)GO TO 400
C      350 CONTINUE
C-----FINAL PRINTOUT-----
C      400 PRINT 511
C      PRINT 512,IT,EMAX,OUT
C      PRINT 513,IT,TRAN,BACK
C-----FORMAT STATEMENTS-----
C      500 FORMAT(1X,12E11.4)
C      502 FORMAT(1X,AMU(J)= X-COSINES*)
C      505 FORMAT(1X,SURF(J), SURFACE SOURCE*)
C      507 FORMAT(1X,A(J), B(J)*
C      509 FORMAT(1X,ANGULAR FLUXES, DIRECTION*,13)
C      511 FORMAT(//)
C      512 FORMAT(1X,NUMBER OF ITERATIONS*,13.5X,MAXIMUM ERROR*,E12.5,2X,
C      1-TOTAL EMERGENT CURRENT*,E12.5)
C      513 FORMAT(1X,CURRENT CONTRIBUTION FOR ITERATION NO.*,19/1X,
C      1-TRANSMISSION*,2X,E13.6/1X,BACKSCATTER*,1X,E13.6)
C      514 FORMAT(1X,TOTAL INCIDENT CURRENT *,E12.5)
C      555 FORMAT(4IS)
C      STOP
C      END
C      SUBROUTINE INPUTS
C      COMMON/ASCAT(24,24,2),AMU(24),SURF(24),A(12,2),B(12,2),EFLUX(101)
C      *CUR(24),S(100,24),SDISTR(100,24),CFUX(100,24),EP(100),X(101),
C      *NGT(24),ERR,ZERO,ETA(2),NX,NANG,NPOS,NSCAT,NORDS,MT,
C      *KSRC,SIG(2),ALBEDO(2),NINT,DX
C      DATA SURF,S,SDISTR/24*0.,4800*0./
C-----READ IN INPUT PARAMETERS-----
C      DX = WIDTH OF X-CELL
C      SIG = 1./COLLISION MEAN FREE PATH
C      ALBEDO = PROBABILITY OF A PARTICLE BEING SCATTERED (NOT
C      ABSORBED).
C      ERR = FRACTIONAL ERROR IN THE SOURCE FUNCTION AT WHICH
C      ITERATIONS ARE TERMINATED.
C      ZERO = LEVEL AT WHICH ERRORS IN THE CALCULATED SOURCE ARE
C      IGNORED
C      NX = NO. OF X-CELLS
C      NPOS = NO. OF POSITIVE DIRECTIONS (=NANG/2)
C      ETA = SCREENING PARAMETER USED WHEN SCREENED-
C      RUTHERFORD SCATTERING IS ASSUMED (NSCAT=1);
C      ALSO ANISOTROPY PARAMETER WHEN NON-ISOTROPIC

```

```

C SCATTERING OTHER THAN SCREENED-RUTHERFORD IS
C ASSUMED.
C SE PARAMETER TO SELECT FORM OF SURFACE SOURCE (
C ANGULAR DEPENDENCE
C = 1. COSINE SOURCE
C = 2. (COSINE)**2 SOURCE
C = 3. (COSINE)**3 SOURCE
C NINT INDEX OF FIRST X-CELL BEYOND MATERIAL INTERFACE.
C IF NO MATERIAL INTERFACE PRESENT, READ IN NINT = 0.

READ 100.DX,ERR,ZERO
READ 10.NX,NPOS,NCASE,NINT
MT=2
IF(NINT.EQ.0)MT=1
DO 1 M=1,MT
1 READ 100.SIG(M),ALBEDO(M),ETA(M)
IF(DX.EQ.0.)DX=.1
IF(ERR.EQ.0.)ERR=.001
IF(ZERO.EQ.0.)ZERO=.0001
C
C NANG=2*NPOS
C PRINT 501.DX,ERR,ZERO,NX,NANG,NPOS,NSCAT,NSRC
C DO 14 M=1,MT
14 PRINT 506.SIG(M),ALBEDO(M)
IF(NSCAT.EQ.0)PRINT 502
IF(NSCAT.EQ.1)PRINT 503,(ETA(M),M=1,MT)
PRINT 505,NCASE,NINT
IF(NINT.EQ.0)NINT=NX+10
C
C COMPUTE GAUSS QUADRATURE ANGLES AND HEIGHTS
C
C N1=NANG+1
C CALL GRULE(NANG,AMU,WGT)
C DO 15 J=1,NPOS
C AMU(N1-J)=AMU(J)
15 WGT(N1-J)=WGT(J)
C
C COMPUTE SCATTERING PROBABILITY FUNCTION
C
C ISOTROPIC SCATTERING
C PROB=0.5
C DO 21 K=1,NANG
C DO 21 L=1,NANG
C DO 21 M=1,MT
C IF(NSCAT.EQ.0) GO TO 21
C SCREENED-RUTHERFORD SCATTERING
C
C AT THIS POINT THE EXPRESSION FOR "PROB", THE SCATTERING
C PROBABILITY(ANGULAR REDISTRIBUTION) FUNCTION CAN BE CHANGED
C ACCORDING TO OTHER SCATTERING MODELS. ANISOTROPIC SCATTERING
C FUNCTIONS OF THE FORM "PROB = 0.5*(1.+ETA(M)*AMU(K)*AMU(L))"
C COULD BE USED, WHERE ETA, THE ANISOTROPY PARAMETER, MAY LIE IN THE
C RANGE 0.0 TO 0.5 (SEE P.205, VOL. 1)
C
C PROB=ETA(M)*(1.+5*ETA(M))*(1.+ETA(M)-AMU(K)*AMU(L))/(ETA(M)*(ETA
C 1(M)*2.+(1.-AMU(K)*AMU(L))+(AMU(K)-AMU(L))**2)**1.5)
21 SCAT(K,L,M)=PROB
C
C SOURCE INITIALIZATION
C IF(NSRC.EQ.1) GO TO 35
C SURFACE (COSINE CURRENT) SOURCE

```

```

DO 30 J=1,NPOS
  C TO (31,32,33),NCASE
  31 IF(2,AMU(J)
  C TO 30
  32 FACTR=3,AMU(J)*AMU(J)
  C TO 30
  33 FACTR=4,AMU(J)*AMU(J)*AMU(J)
  C TO 30
  30 SURF(J)=FACTR
  IF(KSRC.EQ.0)GO TO 40
  35 CONTINUE
C
C CALCULATE SPATIALLY DISTRIBUTED SOURCE FUNCTION, IF APPLICABLE.
C IF THERE IS NO SPATIALLY DISTRIBUTED SOURCE, ZERO OUT THE
C "SDISTR" ARRAY.
C
DO 36 I=1,NX
DO 36 K=1,NANG
  36 SDISTR(I,K)=0.
  40 CONTINUE
C
C-----FORMAT STATEMENTS-----
  10 FORMAT(10I3)
  100 FORMAT(6F10.0)
  501 FORMAT(1X,DX,PF,3,2X,ERR=,
  1F9.6,2X,ZERO,10,7,2X,NH=,13,2X,NANG=,13,2X,NPOS=,13,
  22X,NSCAT=,13,2X,NSRC=,13)
  502 FORMAT(/1X,ISOTROPIC SCATTERING*)
  503 FORMAT(/1X,SCREENED-RUTHERFORD SCATTERING, ETA=,2E12.5)
  505 FORMAT(1X,NCASE = ,13,2X,NINT = ,13)
  506 FORMAT(1X,SIG=,E12.5,2X,ALBEDO=,E12.5)
C
RETURN
END
SUBROUTINE FLUXES(OUT,JS,NANGP,IS,NXP)
C
C THIS SUBROUTINE PERFORMS THE DIAMOND DIFFERENCE ITERATION SCHEME
C OF EQUATIONS 11A, 11B ON P. 198, VOL. I.
C
COMMON/A/SCAT(24,24,2),AMU(24),SURF(24),A(12,2),B(12,2),EFLUX(101)
+.CUR(24),S(100,24),SDISTR(100,24),CFUX(100,24),EF(100),X(101),
+WGT(24),ERR,ZERO,ETA(2),NX,NANG,NPOS,NSCAT,NORDS,MT,
+KSRC,SIG(2),ALBEDO(2),NINT,DX
C
IF(JS.NE.1)GO TO 10
NXP1=NXP+1
DO 15 I=1,NXP1
  15 EFLUX(I)=0.
  10 DO 100 J=1,NPOS
    JR=J+JS+NANGP
    EF(J)=SURF(JR)
    IF(15.EQ.-1)EFLUX(NXP1)=EFLUX(NXP1)+WGT(J)*EF(J)
    DO 90 I=1,NX
      IF(15.EQ.-1)GO TO 20
      M=1
      IF(1.GE.NINT)M=2
      GO TO 21
    20 M=2
    IF(1.LT.NINT)M=1
    21 IR=I+IS+NXP
    IF(15.EQ.1)EFLUX(IR)=EFLUX(IR)+WGT(J)*EF(J)
    CFUX(IR,JR)=A(J,M)*EF(J)+B(J,M)*S(IR,JR)
    P=2.-CFUX(IR,JR)
    EF(J)=P-EF(J)
    IF(15.EQ.-1)EFLUX(IR)=EFLUX(IR)+WGT(J)*EF(J)
  90 CONTINUE

```


Program Listing

- BEAMSRC -

(Reference: Vol. I, Section IV.A.2)

```

PROGRAM BEAMSRC(INPUT,OUTPUT,TAPE7,TAPE8)

C BEAMSRC CALCULATES, BY MEANS OF THE METHOD OF DISCRETE ORIGINATES,
C THE FLUX DUE TO A ONCE SCATTERED SOURCE OF ELECTRONS FROM
C MONODIRECTIONAL BEAM INCIDENT ON A SLAB. BOTH SCREENED RUTHERFORD
C AND ISOTROPIC SCATTERING ARE TREATED. WHEN SCREENED
C RUTHERFORD SCATTERING IS ASSUMED, EXTENDED TRANSPORT CORRECTED
C CROSS SECTIONS ARE EMPLOYED. NEGATIVE FLUX FIXUP IS AND COARSE
C MESH REBALANCE CONVERGENCE ACCELERATION ARE INCLUDED.

COMMON SCAT(12,12), ANU(12), A(12,2), B(12,2), EFUX( 50), ESRC
+CUR(12), S( 5), L2) SDISTR( 50), CFUX( 50), Z( 2), EF( 50), X( 51),
+NOT(12), DX, SIG(2), ALBEDO(2), ERR, ZERO, ETA(2), NX, NANG, NPOS,
+KSRG, INTRFC, NLAYE, L2(2), AA(2), IPRNT, COSTN, ENTAB(100), DS(2),
+DISTR(13,2), LAMPR(2), LMAA, PL(13,12), ISO, IEDGE, ISKIP, DELE(50),
+XCIT(21,2), STPNR(2), DE, IITER(2), NGRPS, NG, ITMAX, IT, MATL(2), CSUM,
+CMESH(2,10), FBAL(10), INBL, SUM1, NCRS, STPO, ELAMO

C DIMENSION BCUR(12), EDPOS(100), CONT(100), FJ(51), BU(51), TOTJ(51)
C DIMENSION EFOLD(51)
C DATA EFOLD/51*0./
C DATA EDPOS,CONT/200*0./

C-----INPUT PARAMETERS-----
C ITMAX = MAXIMUM NO. OF ITERATIONS
C ISO = 0/1, EDGE ANGULAR FLUX COMPUTATION, NO/YES
C ISKIP = FLUX PRINTOUT AT EVERY ISKIP-TM SPATIAL POSITION
C INBL = 0/1, COARSE MESH REBALANCE OPTION, YES/NO
C NCRS = NUMBER OF COARSE MESH INTERVALS
C NCRSKP = COARSE MESH REBALANCE(CMR) SKIP PARAMETER
C (PERFORM CMR EVERY NCRSKP ITERATIONS)
C DX = WIDTH OF X-CELL(GM-CM**2)
C NGRPS = NUMBER OF ENERGY GROUPS
C IPRNT = PRINTOUT PARAMETER,0/1/2/3 = FINAL SCALAR FLUX ONLY/
C ITERATIONS/ANGULAR FLUX/FULL DEBUG PRINTOUT
C SIG(M) = TOTAL INTERACTION CROSS SECTION FOR MATERIAL M.
C UNITS ARE (CM**2/CM)
C ALBEDO(M) = PROBABILITY OF A PARTICLE BEING SCATTERED (NOT
C ABSORBED) FOR MATERIAL M
C ERR = FRACTIONAL ERROR IN THE SOURCE FUNCTION AT WHICH
C ITERATIONS ARE TERMINATED.
C ZERO = LEVEL AT WHICH ERRORS IN THE CALCULATED SOURCE ARE
C IGNORED
C NX = NO. OF X-CELLS
C NPOS = NO. OF POSITIVE DIRECTIONS
C ETA(M) = SCREENING PARAMETER FOR MATERIAL M USED WHEN
C SCREENED RUTHERFORD SCATTERING IS ASSUMED
C INTRFC = INDEX OF MATERIAL BOUNDARY INTERFACE
C E = ELECTRON KINETIC ENERGY(MEV)
C DE = ENERGY GROUP WIDTH (MEV)
C COSIN = COSINE OF SOURCE BEAM ANGLE OF INCIDENCE
C Z(M) = COSINE OF SOURCE BEAM ANGLE OF INCIDENCE
C AA(M) = ATOMIC NUMBER OF MATERIAL M
C = ATOMIC WEIGHT OF MATERIAL M

C-----DERIVED PARAMETERS-----
C SCAT(K,L,M) = SCATTERING KERNEL FOR MATERIAL M
C NANG = TOTAL NO. OF DISCRETE ORIGINATES (ANGLES)=2*NPOS
C ANU(J) = DIRECTION COSINES OF POSITIVE DIRECTIONS

```

```

C      SDISTR(1) = DISTRIBUTED SOURCE AT CENTER OF ITH CELL
C      (ACTUALLY SDISTR(1,K) WHERE K DENOTES THE K-TH
C      DIRECTION. THE SDISTR(1) ARRAYS ARE STORED IN K
C      RECORDS OF LENGTH NX ON SCRATCH FILE TAPE7).
C-----CALCULATED QUANTITIES-----
C      CFUX(1,K) = ANGULAR FLUX AT CENTER OF ITH CELL IN KTH DIRECTION
C      EF(K) = ANGULAR FLUX AT CELL EDGE
C      FLUX(1) = SCALAR FLUX AT CELL EDGE
C      CUR(K) = PARTICLE CURRENT IN KTH DIRECTION
C      S(1,K) = SCATTERED SOURCE FUNCTION
C      OUT = TOTAL EMERGENT CURRENT FROM SLAB
C      ETA(M) = ATOMIC(RUTHERFORD) SCREENING CONSTANT FOR MATERIAL
C      M. IT IS CALCULATED FROM THE MOLIERE FORMULA
C      SGG(L) = LEGENDRE COEFFICIENTS OF SCATTERING CROSS SECTION
C      (UNCORRECTED)
C      SIGSTR(L) = LEGENDRE COEFFICIENTS OF TRANSPORT CORRECTED
C      SCATTERING CROSS SECTION
C      SIGA = ABSORPTION CROSS SECTION (=ABSORB*SGG(1))
C      DLSTR(L,M) = LEGENDRE COEFFICIENTS OF TRANSPORT CORRECTED
C      SCATTERING PROBABILITY
C      STPMR(M) = ELECTRON STOPPING POWER FOR MATERIAL M.
C      UNITS ARE (MEV-CM**2/GM)
C      DS(M) = ELECTRON PATH LENGTH IN MATERIAL M FOR ENERGY
C      GROUP OF WIDTH DE.
C      UNITS ARE (GM/CM**2)
C-----BEGIN CALCULATIONS-----
C      DATA ETAPM,EBACK,TSUM,BSUM/A*0. /
C      READ 555,NRUN,ITMAX,NCRPS,IPRNT,ISO,IEDGE,ISKIP,IRBL,NCRS,NCRSKP
C      PRINT 557,NRUN
C      PRINT 556,ITMAX,NCRPS,IPRNT,ISO,IEDGE,ISKIP,IRBL,NCRS,NCRSKP
C
C      SET NG, THE ENERGY GROUP INDEX
C      CALL INPUTS
C      NG=1
C      ECUM=ESRC
C      IF(IPRNT.LT.3)GO TO 60
C-----PRINT OUT DERIVED CONSTANTS, BOUNDARY CONDITIONS, ETC.-----
C      PRINT 511
C      PRINT 502
C      PRINT 500,(AMU(J),J=1,NPOS)
C
C      PRECOMPUTATION OF A AND B ARRAYS (USED SEVERAL TIMES IN SUBROUTINE
C      FLUX
C
C      60 DO 100 J=1,NPOS
C      DO 100 M=1,NLAY
C      A(J,M)=2.*AMU(J)/(2.*AMU(J)+SIG(M)*DX)
C      100 B(J,M)=DX/(2.*AMU(J)+SIG(M)*DX)
C      IF(IPRNT.LT.3) GO TO 61
C      DO 101 M=1,NLAY
C      PRINT 511
C      PRINT 507,M
C      PRINT 500,(A(J,M),J=1,NPOS)
C      PRINT 511
C      PRINT 508,M
C      101 PRINT 500,(B(J,M),J=1,NPOS)
C
C      INITIALIZE DISTRIBUTED SOURCE
C      61 REWIND 7
C      DO 13 K=1,NANG
C      READ(7)(SDISTR(1),I=1,NX)
C      DO 13 II=1,NX
C      13 S(11,K)=SDISTR(II)

```



```

C-----
C      11 M=1,NANG
C      DO 11 I=1,NX
C      11 CFLUX(I,K)=0.
C
C      COMPUTE COORDINATES, X(I), OF CELL EDGES (NXP OF THEM)
C
C      NXP=NX+1
C      DO 12 I=1,NXP
C      FJ(I)=BU(I)*TOTJ(I)=0.
C      12 X(I)=(I-1)*DX
C      XMIN=0.
C      XMAX=X(NXP)
C      YMIN=0.
C
C      TOTAL NUMBER OF ANGLES = NANG
C
C      NANG=2*NPOS
C
C      REMINDO PLOT FILE BUFFER
C
C      REMIND 8
C      WRITE(8)NRUN,NGRPS,NXP,NANG,(X(I),I=1,NXP)
C
C      ENERGY GROUP (OUTER ITERATION) LOOP
C
C      1000 CONTINUE
C      IX=NX/NCRS
C      PRINT 511
C-----MAIN (INNER) ITERATION LOOP-----
C      DO 350 IT=1,ITMAX
C      IF(NG.LT.3)GO TO 699
C      IRBL=0
C      IF(IT.LE.10)GO TO 699
C      IF((IT/NCRSKP)*NCRSKP-IT).NE.0)IRBL=1
C      699 OUT=TRAN=BACK=0.
C      CALL FLUXES(TRAN,1,0,1,0)
C      CALL FLUXES(BACK,-1,LMAX,-1,NXP)
C      IF(IRBL.GT.0)GO TO 715
C
C      PERFORM COARSE MESH REBALANCE
C
C      CALL REBAL
C      DO 710 J=1,NANG
C      DO 710 I=1,NX
C      IRR=(1+IX-1)/IX
C      710 CFLUX(I,J)=CFLUX(I,J)*FBAL(IRR)
C      TRAN=TRAN+FBAL(NCRS)
C      BACK=BACK+FBAL(1)
C      IF(IT.EQ.1)PRINT 885,IT,(FBAL(1),I=1,NCRS)
C      715 OUT=TRAN+BACK
C
C-----CALCULATE NEW SOURCE-----
C      ENAX=0.
C      REMIND 7
C      DO 200 K=1,NANG
C      READ(7)(SDISTR(I),I=1,NX)
C      DO 200 I=1,NX
C      SUM=SDISTR(I)
C      M=1
C      IF(I.GE.INTRFC)M=2
C      DO 150 L=1,NANG
C      150 SUM=SUM+SCAT(K,L,M)*CFLUX(I,L)*WG(L)*ALBEDO(M)*SIG(M)
C      IF(ABS(SUM).LT.ZERO)GO TO 200
C      200 S(I,K)=SUM

```

```

2200 I=1,NXP
   FLUX(I),EQ.0.0)GO TO 2200
   E=OR-ABS(1.-EFLD(I)/EFLUX(I))
   IF(ERROR.GT.EMAX)EMAX=ERROR
2200 EFLD(I)=EFLUX(I)
   IF(I.PRINT.LT.2) GO TO 83
   PRINT 512,IT,EMAX.OUT,TRAN,BACK
63 IF(EMAX.LT.ERR)GO TO 400
350 CONTINUE
C
C-----FINAL PRINTOUT FOR THE NG-TH ENERGY GROUP-----
400 PRINT 505,IT
   IF(I.PRINT.LT.2)PRINT 512,IT,EMAX.OUT,TRAN,BACK
   IF(I.PRINT.LT.2)PRINT 885,IT,(FVAL(I),I=1,NCRS)
885 FORMAT(1X,ITERATION=,IS,REBALANCE FACTORS=,/(10E12.5))
   ETRAN=ETRAN+TRAN+DELE(NG)
   EBACK=EBACK+BACK+DELE(NG)
   TSUM=TSUM+TRAN
   BSUM=BSUM+BACK
   CSUM=OUT
   DO 62 J=1,NPOS
62 BCUR(J)=CUR(LMAX-J)
   IF(I.PRINT.EQ.1)PRINT 561,(CUR(J),J=1,NPOS)
   DO 62 J=1,NPOS
62 BCUR(J)=CUR(LMAX-J)
   IF(I.PRINT.EQ.1)PRINT 562,(BCUR(J),J=1,NPOS)
C
C PRINT CELL CENTER ANGULAR FLUXES
C
   IF(I.PRINT.LT.2)GO TO 64
   K1=-7
395 K1=K1+8
   K2=K1+7
   IF(K2.GT.NANG)K2=NANG
   PRINT 509
   DO 300 I=1,NX,ISKIP
300 PRINT 517,I,(CFLUX(I,K),K=K1,K2)
   IF(K2.LT.NANG)GO TO 395
C COMPUTE EDGE ANGULAR FLUXES(ONLY VALID WHEN DIAMOND DIFF USED)
C (THE SOURCE ARRAY IS BORROWED HERE FOR THE EDGE FLUX COMPUTATION)
64 DO 579 J=1,NANG
   DO 579 I=1,NXP
579 S(I,J)=0.0
   DO 580 J=1,NPOS
   DO 580 I=2,NXP
580 S(I,J)=2.*CFLUX(I-1,J)-S(I-1,J)
   DO 581 J=1,NPOS
   JJ=LMAX-J
   DO 581 I=1,NX
   II=NXP-I
581 S(II,JJ)=2.*CFLUX(II,JJ)-S(II+1,JJ)
   IF(IEEDGE.EQ.0)GO TO 380
   K1=-7
396 K1=K1+8
   K2=K1+7
   IF(K2.GT.NANG)K2=NANG
   PRINT 510
C PRINT CELL EDGE ANGULAR FLUXES
   DO 306 I=1,NXP,ISKIP
306 PRINT 517,I,(S(I,K),K=K1,K2)
   IF(K2.LT.NANG)GO TO 396
380 CONTINUE
C
C COMPUTE ENERGY FLOW
   DO 370 I=1,NXP
   DO 371 J=1,NPOS
   JJ=LMAX-J
   FU(I)=FU(I)+WGT(J)*AMU(J)*S(I,J)

```

```

371 B(1)=BJ(I)+WGT(JJ)*AMU(JJ)*S(I,JJ)
370 I(1)=FJ(I)+BJ(I)
DO 381 I=1,NXP
DO 381 K=1,NANG
381 S(I,K)=0.
C COMPUTE ENERGY DEPOSITION PROFILE CONTRIBUTION FROM SCATTERED
C ELECTRONS
M=1
DO 450 I=1,NX
CONT(I)=0.
IF(I.GE.INTRFC)M=2
DO 449 J=1,NANG
449 CONT(I)=CONT(I)+STPWR(M)*WGT(J)*CFUX(I,J)
450 CONTINUE
PRINT 511
PRINT 515
C THE EFLUX ARRAY IS BORROWED HERE TO COMPUTE THE UNSCATTERED
C FLUX CONTRIBUTION. THE SOURCE WAS THE ONCE SCATTERED
C DISTRIBUTED SOURCE RESULTING FROM THE UNSCATTERED
C MONODIRECTIONAL BEAM. THE UNSCATTERED FLUX CONTRIBUTION IS
C COMPUTED FOR THE FIRST(SOURCE) ENERGY GROUP ONLY.
IF(NG.GT.1)GO TO 309
DO 382 I=1,NXP
382 S(I,2)=EFLUX(I)
YMAX=1.25
C
THK=DX*(INTRFC-1)
M=1
SAVE=0.
CC=STPD*ELAMO*COSIN/DX
CD=1./(ELAMO*COSIN)
DO 308 I=1,NX
IF(I.GE.INTRFC)M=2
IF(INTRFC.EQ.NXP)M=1
IF(I.GT.INTRFC)GO TO 307
S(I,1)=SAVE+EXP(-X(I)*SIG(M)/COSIN)
EDPOS(I)=CC*EXP(-CD*X(I))-EXP(-CD*X(I+1))
EFLUX(I)=EFLUX(I)+SAVE
FJ(I)=FJ(I)+COSIN*SAVE
GO TO 308
307 S(I,1)=SAVI+SAVE*EXP(-(X(I)-THK)*SIG(2)/COSIN)
EFLUX(I)=EFLUX(I)+SAVI
308 CONTINUE
PRINT 5663,(EDPOS(I),I=1,NX,ISKIP)
5663 FORMAT(1X,'ENERGY DEPOSITION PROFILE CONTRIBUTION DUE TO UNSCATE
RED FLUX',(10E12.5))
TR=S(NXP,1)*COSIN
BK=0.
WRITE(8)ESRC,TR,BK,YMIN,YMAX,(S(I,1),I=1,NXP)
YMAX=0.
DO 378 I=1,NXP
378 IF(S(I,2).GE.YMAX)YMAX=S(I,2)
WRITE(8)ENTAB(NG),TRAN,BACK,YMIN,YMAX,(S(I,2),I=1,NXP)
ETRA=ETRA+COSIN*S(NXP,1)*DELE(1)
C PRINT CELL EDGE SCALAR FLUXES
C
309 PRINT 516,(I,X(I),EFLUX(I),I=1,NXP,ISKIP)
ETL=0.
DO 319 I=1,NX
319 EDPOS(I)=EDPOS(I)+CONT(I)
ETL=ETL+EDPOS(I)
PRINT 1748,ETL
C
C WRITE PLOT FILE

```

```

C      ENID=ENTAB(NG)
C      I=NG.GT.1)WRITE(8)EMID,TRAN,BACK,YMIN,YMAX,(EFLUX(I),I=1,NX)
C
C      PRINT THE ENERGY DEPOSITION PROFILES
C
      IF(I.PRINT.GE.1)PRINT 563,(CONT(I),I=1,NX,ISKIP)
      IF(I.PRINT.GE.1)PRINT 564,(EDPOS(I),I=1,NX,ISKIP)
      ECUM=ECUM-DELE(NG)
      IF(NG.LT.(NGRPS-5))GO TO 1752
      DO 1750 I=1,NX
1750  EFLUX(I)=0.5*(EFLUX(I)+EFLUX(I+1))*ECUM/DX+EDPOS(I)
      PRINT 1751,(EFLUX(I),I=1,NX)
1751  FORMAT(1X,'TOTAL ENERGY DEPOSITION PROFILE IF CALCULATION STOPS
      & HERE'/(10E12.5))
      ETL=0.
      DO 1749 I=1,NX
1749  ETL=ETL+DX+EFLUX(I)
      PRINT 1748,ETL
1748  FORMAT(1X,'TOTAL ENERGY DEPOSITED (MEV) =',E12.5)
1752  CONTINUE
C
C      DECREMENT THE ELECTRON ENERGY TO THE NEXT GROUP
C
      NG=NG+1
      E=ENTAB(NG)
      IF(NG.LE.NGRPS)GO TO 600
      EFLMIN=1.E10
      EFLMAX=0.
      DO 602 I=1,NXP
602  IF(TOTJ(I).GE.EFLMAX)EFLMAX=TOTJ(I)
      IF(TOTJ(I).LE.EFLMIN)EFLMIN=TOTJ(I)
      WRITE(8)EFLMIN,EFLMAX,(TOTJ(I),I=1,NXP)
      PRINT 570,(TOTJ(I),I=1,NXP)
      PRINT 571,(F(I),I=1,NXP)
      PRINT 572,(B(I),I=1,NXP)
      PRINT 573,ETRAN,EBACK
1753  FORMAT(1X,'TOTAL ENERGY TRANSMISSION=',E12.5,'MEV',1X,
      & 11X,'TOTAL ENERGY BACKSCATTER=',E12.5,'MEV')
1754  FORMAT(1X,'NET CELL EDGE CURRENTS'/(10E12.5))
1755  FORMAT(1X,'FORWARD CELL EDGE CURRENTS'/(10E12.5))
1756  FORMAT(1X,'BACKWARD CELL EDGE CURRENTS'/(10E12.5))
      EDMAX=0.
      DO 601 I=1,NX
601  X(I)=X(I)+5*DX
      EDPOS(I)=EDPOS(I)+1000.
      IF(EDPOS(I).GE.EDMAX)EDMAX=EDPOS(I)
601  CONTINUE
      WRITE(8)NX,EDMAX,(X(I),I=1,NX),(EDPOS(I),I=1,NX)
      GO TO 610
C
C      COMPUTE NEW CROSS SECTIONS, SCATTERING PROBABILITY, AND
C      SCATTERING SOURCE
C
      GO 600 CALL RECALC
      GO TO 1000
C
C-----FORMAT STATEMENTS
500  FORMAT(1X,12E11.4)
502  FORMAT(1X,'ANU(J)= X-COSINES')
505  FORMAT(1X,'TOTAL NUMBER OF ITERATIONS = ',I5)
507  FORMAT(1X,'A - ARRAY FOR MATERIAL NO.',I2)
508  FORMAT(1X,'B - ARRAY FOR MATERIAL NO.',I2)
509  FORMAT(1X,'ANGULAR CENTER FLUXES'/(1X,'CELL NO.))

```

```

511 7  'AT(//)
512 7  'AT(1X,.1ITERATION*.13,. MAX. ERROR=.E12.5.
      *+ EXIT CURRENT=.E12.5.
      *+ TRANSMISSION=.E12.5. BACKSCATTER=.E12.5)
513 7  'FORMAT(1X,.CURRENT CONTRIBUTION FOR ITERATION NO.+.15/1X.
      1-TRANSMISSION+.2X.E13.6/1X.*BACKSCATTER+.1X.E13.6)
514 7  'FORMAT(1X,.TOTAL INCIDENT CURRENT *.E12.5)
515 7  'FORMAT(1X,.TOTAL SCALAR EDGE FLUX AT X(I)*)
      +2X.*1.-8X.*X(I)*)+7X.*FLUX(I)*)
516 7  'FORMAT(4(1X.13.2X.2E12.5))
517 7  'FORMAT(1X.14.5X.8E12.5)
518 7  'FORMAT(10I5)
519 7  'FORMAT(1X,.1TMAX*.15. (MAXIMUM ALLOWED NUMBER OF INNER ITERATION
      *+S)/1X.*NCRPS*.15. (NUMBER OF ENERGY GROUPS)/1X.*IPRNT*.15. P
      *RINTOUT CONTROL (0/1/2/3 = FINAL SCALAR FLUX ONLY/ITERATIONS/ANGUL
      *AR FLUX/FULL DEBUG PRINTOUT)/1X.*ISO *.15. (0/1. SCREENED RUTH
      *ERFORD SCATTERING/ISOTROPIC SCATTERING)/1X.*EDGE*.15.
      *+ (0/1. ANGULAR EDGE FLUX COMPUTATION, NO/YES)*)
      +1X.*ISKIP*.15. (FLUX PRINTOUT PARAMETER - PRINT EVERY ISKIP-TH C
      *+EL)/1X.*IRBL*.15. (0/1. COARSE MESH REBALANCE, YES/NO)*)
      +1X.*NCRS*.15. (NUMBER OF COARSE MESH INTERVALS)/1X.*NCRSHP *.
      +15. COARSE MESH REBALANCE SKIP PARAMETER*)
520 7  'FORMAT(1X,.RUN NUMBER *.14)
521 7  'FORMAT(1X,.ANGULAR EDGE FLUXES*/1X.*POINT NO.*)
522 7  'FORMAT(1X,.ANGULAR TRANSMISSION FRACTION (AT GAUSSIAN ORDINATES)*
      1/(10E12.5))
523 7  'FORMAT(1X,.ANGULAR BACKSCATTER FRACTION (AT GAUSSIAN ORDINATES)*
      1/(10E12.5))
524 7  'FORMAT(1X,.ENERGY DEPOSITION PROFILE CONTRIBUTION FOR THIS GROUP*
      1/(10E12.5))
525 7  'FORMAT(1X,.CUMULATIVE ENERGY DEPOSITION PROFILE/(10E12.5))
526 7  'FORMAT(1X,.TRANSMISSION=.E12.5.5X.*BACKSCATTER=.E12.5)
527 7  'STOP
528 7  'END
529 7  'SUBROUTINE INPUTS
530 7  'COMMON SCAT(12,12,2).AMU(12).A(12,2).B(12,2).EFLUX( 51).ESRC
      *+CUR(12).S( 51,12).SDISTR( 50).CFUX( 50,12).EF( 50).X( 51).
      *+WGT(12).DX.SIG(2).ALBEDO(2).ERR,ZERO,ETA(2).NX.NANG.NPOS.
      *+KSRC,INTRFC,NLAY,E,Z(2).AA(2).IPRNT.COSIN.ENTAB(100).DS(2).
      *+DLSTR(13,2).ELAMPRI(2).LMAX.PL(13,12).ISO.EDGE.ISKIP.DELE(50).
      *+EXCIT(2).STPR(2).DE.ITER(2).NCRPS.NG.ITMAX.IT.MATL(2).CSUM.
      *+CMESH(2,10).FBAL(10).IRBL.SUM1.NCRS.STPO.ELANO
531 7  'DIMENSION THK(2).Y(25)
532 7  'DIMENSION ZM(3).AM(3)
533 7  'DATA ZM,AM/13..29..79..26.98,63.54,197./
534 7  'Y(N) = OUTPUT ARRAY FROM SUBROUTINE LEP
535 7  'PL(N,K) = LEGENDRE POLYNOMIAL OF ORDER (N-1) FOR THE K-TH
      GAUSSIAN QUADRATURE ANGLE
536 7  'ZERO OUT DISTRIBUTED SOURCE, CROSS SECTION, ALBEDO
      AND SCREENING FACTOR ARRAYS
537 7  'DO 101 J=1,100
538 7  'SDISTR(J)=0.
539 7  'DO 101 I=1,12
540 7  'S(J,I)=0.
541 7  'DO 102 M=1,2
542 7  '102 SIG(M)=ALBEDO(M)=ETA(M)=0.
543 7  'C
544 7  'C
545 7  'C
546 7  'C
547 7  'C
548 7  'C
549 7  'C
550 7  'C
551 7  'C
552 7  'C
553 7  'C
554 7  'C
555 7  'C
556 7  'C
557 7  'C
558 7  'C
559 7  'C
560 7  'C
561 7  'C
562 7  'C
563 7  'C
564 7  'C
565 7  'C
566 7  'C
567 7  'C
568 7  'C
569 7  'C
570 7  'C
571 7  'C
572 7  'C
573 7  'C
574 7  'C
575 7  'C
576 7  'C
577 7  'C
578 7  'C
579 7  'C
580 7  'C
581 7  'C
582 7  'C
583 7  'C
584 7  'C
585 7  'C
586 7  'C
587 7  'C
588 7  'C
589 7  'C
590 7  'C
591 7  'C
592 7  'C
593 7  'C
594 7  'C
595 7  'C
596 7  'C
597 7  'C
598 7  'C
599 7  'C
600 7  'C
601 7  'C
602 7  'C
603 7  'C
604 7  'C
605 7  'C
606 7  'C
607 7  'C
608 7  'C
609 7  'C
610 7  'C
611 7  'C
612 7  'C
613 7  'C
614 7  'C
615 7  'C
616 7  'C
617 7  'C
618 7  'C
619 7  'C
620 7  'C
621 7  'C
622 7  'C
623 7  'C
624 7  'C
625 7  'C
626 7  'C
627 7  'C
628 7  'C
629 7  'C
630 7  'C
631 7  'C
632 7  'C
633 7  'C
634 7  'C
635 7  'C
636 7  'C
637 7  'C
638 7  'C
639 7  'C
640 7  'C
641 7  'C
642 7  'C
643 7  'C
644 7  'C
645 7  'C
646 7  'C
647 7  'C
648 7  'C
649 7  'C
650 7  'C
651 7  'C
652 7  'C
653 7  'C
654 7  'C
655 7  'C
656 7  'C
657 7  'C
658 7  'C
659 7  'C
660 7  'C
661 7  'C
662 7  'C
663 7  'C
664 7  'C
665 7  'C
666 7  'C
667 7  'C
668 7  'C
669 7  'C
670 7  'C
671 7  'C
672 7  'C
673 7  'C
674 7  'C
675 7  'C
676 7  'C
677 7  'C
678 7  'C
679 7  'C
680 7  'C
681 7  'C
682 7  'C
683 7  'C
684 7  'C
685 7  'C
686 7  'C
687 7  'C
688 7  'C
689 7  'C
690 7  'C
691 7  'C
692 7  'C
693 7  'C
694 7  'C
695 7  'C
696 7  'C
697 7  'C
698 7  'C
699 7  'C
700 7  'C
701 7  'C
702 7  'C
703 7  'C
704 7  'C
705 7  'C
706 7  'C
707 7  'C
708 7  'C
709 7  'C
710 7  'C
711 7  'C
712 7  'C
713 7  'C
714 7  'C
715 7  'C
716 7  'C
717 7  'C
718 7  'C
719 7  'C
720 7  'C
721 7  'C
722 7  'C
723 7  'C
724 7  'C
725 7  'C
726 7  'C
727 7  'C
728 7  'C
729 7  'C
730 7  'C
731 7  'C
732 7  'C
733 7  'C
734 7  'C
735 7  'C
736 7  'C
737 7  'C
738 7  'C
739 7  'C
740 7  'C
741 7  'C
742 7  'C
743 7  'C
744 7  'C
745 7  'C
746 7  'C
747 7  'C
748 7  'C
749 7  'C
750 7  'C
751 7  'C
752 7  'C
753 7  'C
754 7  'C
755 7  'C
756 7  'C
757 7  'C
758 7  'C
759 7  'C
760 7  'C
761 7  'C
762 7  'C
763 7  'C
764 7  'C
765 7  'C
766 7  'C
767 7  'C
768 7  'C
769 7  'C
770 7  'C
771 7  'C
772 7  'C
773 7  'C
774 7  'C
775 7  'C
776 7  'C
777 7  'C
778 7  'C
779 7  'C
780 7  'C
781 7  'C
782 7  'C
783 7  'C
784 7  'C
785 7  'C
786 7  'C
787 7  'C
788 7  'C
789 7  'C
790 7  'C
791 7  'C
792 7  'C
793 7  'C
794 7  'C
795 7  'C
796 7  'C
797 7  'C
798 7  'C
799 7  'C
800 7  'C
801 7  'C
802 7  'C
803 7  'C
804 7  'C
805 7  'C
806 7  'C
807 7  'C
808 7  'C
809 7  'C
810 7  'C
811 7  'C
812 7  'C
813 7  'C
814 7  'C
815 7  'C
816 7  'C
817 7  'C
818 7  'C
819 7  'C
820 7  'C
821 7  'C
822 7  'C
823 7  'C
824 7  'C
825 7  'C
826 7  'C
827 7  'C
828 7  'C
829 7  'C
830 7  'C
831 7  'C
832 7  'C
833 7  'C
834 7  'C
835 7  'C
836 7  'C
837 7  'C
838 7  'C
839 7  'C
840 7  'C
841 7  'C
842 7  'C
843 7  'C
844 7  'C
845 7  'C
846 7  'C
847 7  'C
848 7  'C
849 7  'C
850 7  'C
851 7  'C
852 7  'C
853 7  'C
854 7  'C
855 7  'C
856 7  'C
857 7  'C
858 7  'C
859 7  'C
860 7  'C
861 7  'C
862 7  'C
863 7  'C
864 7  'C
865 7  'C
866 7  'C
867 7  'C
868 7  'C
869 7  'C
870 7  'C
871 7  'C
872 7  'C
873 7  'C
874 7  'C
875 7  'C
876 7  'C
877 7  'C
878 7  'C
879 7  'C
880 7  'C
881 7  'C
882 7  'C
883 7  'C
884 7  'C
885 7  'C
886 7  'C
887 7  'C
888 7  'C
889 7  'C
890 7  'C
891 7  'C
892 7  'C
893 7  'C
894 7  'C
895 7  'C
896 7  'C
897 7  'C
898 7  'C
899 7  'C
900 7  'C
901 7  'C
902 7  'C
903 7  'C
904 7  'C
905 7  'C
906 7  'C
907 7  'C
908 7  'C
909 7  'C
910 7  'C
911 7  'C
912 7  'C
913 7  'C
914 7  'C
915 7  'C
916 7  'C
917 7  'C
918 7  'C
919 7  'C
920 7  'C
921 7  'C
922 7  'C
923 7  'C
924 7  'C
925 7  'C
926 7  'C
927 7  'C
928 7  'C
929 7  'C
930 7  'C
931 7  'C
932 7  'C
933 7  'C
934 7  'C
935 7  'C
936 7  'C
937 7  'C
938 7  'C
939 7  'C
940 7  'C
941 7  'C
942 7  'C
943 7  'C
944 7  'C
945 7  'C
946 7  'C
947 7  'C
948 7  'C
949 7  'C
950 7  'C
951 7  'C
952 7  'C
953 7  'C
954 7  'C
955 7  'C
956 7  'C
957 7  'C
958 7  'C
959 7  'C
960 7  'C
961 7  'C
962 7  'C
963 7  'C
964 7  'C
965 7  'C
966 7  'C
967 7  'C
968 7  'C
969 7  'C
970 7  'C
971 7  'C
972 7  'C
973 7  'C
974 7  'C
975 7  'C
976 7  'C
977 7  'C
978 7  'C
979 7  'C
980 7  'C
981 7  'C
982 7  'C
983 7  'C
984 7  'C
985 7  'C
986 7  'C
987 7  'C
988 7  'C
989 7  'C
990 7  'C
991 7  'C
992 7  'C
993 7  'C
994 7  'C
995 7  'C
996 7  'C
997 7  'C
998 7  'C
999 7  'C
1000 7  'C

```

```

C      J 10,NX,NPOS,INTRFC
C      IF THERE IS NO MATERIAL INTERFACE (ONLY ONE SCATTERING
C      MATERIAL), INTRFC IS READ IN AS "0". IT IS THEN SET TO
C      NX+1 (OUTSIDE THE RIGHT BOUNDARY).
C      NLAY IS THE NUMBER OF MATERIAL LAYERS.
C      IF(INTRFC.EQ.0)INTRFC=NX+1
C      NLAY=1
C      IF(INTRFC.LT.NX.AND.INTRFC.GT.1)NLAY=2
C      FILL IN THE ATOMIC NUMBER (Z), ATOMIC WEIGHT (AA), AND MEAN
C      EXCITATION ENERGY(EXCIT) ARRAYS
C      READ 10,(MATL(I),I=1,NLAY)
C      DO 9 I=1,NLAY
C      MM=MATL(I)
C      EXCIT(I)=(9.76*ZM(MM)+58.5/(ZM(MM)**.19))*1.E-6
C      AA(I)=AM(MM)
C      Z(I)=ZM(MM)
C      9 Z(I)=ZM(MM)
C      DX = SPATIAL CELL WIDTH (CM/CM**2)
C      ERR = TOLERANCE FOR ITERATION CONVERGENCE
C      ZERO = "SAFE" VALUE FOR FLOATING POINT ZERO
C      READ 100,DX,ERR,ZERO
C      E = ELECTRON SOURCE ENERGY (GROUP 1) IN MEV
C      DE = ENERGY GROUP WIDTH IN MEV
C      READ 100,E
C      ESRC=E
C      READ COSINE OF INCIDENT BEAM
C      READ 100,COSIN
C      DEFAULT VALUES FOR DX, ERR, ZERO, COSIN
C      IF(DX.EQ.0.)DX=.01
C      IF(ERR.EQ.0.)ERR=.0001
C      IF(ZERO.EQ.0.)ZERO=.00001
C      IF(COSIN.EQ.0.)COSIN=1.0
C      IF SCATTERING IS ISOTROPIC, READ IN TOTAL CROSS
C      SECTION (SIG) AND ALBEDO (ALBEDO) FOR EACH MATERIAL.
C      IF(ISO.GT.0)READ 100,(SIG(M),ALBEDO(M),M=1,NLAY)
C      NANG=2*NPOS
C      PRINT 501,DX,ERR,ZERO,NX,NANG,NPOS
C      COMPUTE GAUSS QUADRATURE ANGLES AND WEIGHTS
C      LMAX=NANG+1
C      CALL GRULE(NANG,AMU,WGT)
C      DO 15 J=1,NPOS
C      AMU(LMAX-J)=AMU(J)
C      WGT(LMAX-J)=WGT(J)
15  IF(IPRNT.LT.1)GO TO 17
C      PRINT 508,(AMU(J),J=1,NPOS)
C      PRINT 509,(WGT(J),J=1,NPOS)

```



```

11=1
X1 0.0
D1 / I=1,NX
M=1
IF(I-INTRFC)33,32,31
31 M=2
GO TO 33
32 XAS=XX
I=INTRFC
33 XX=(I-IT)*DX/ELAMPR(M)*XXS
FAC=ALBEDO(M)/ELAMPR(M)*EXP(-XX/COSIN)
S(I,K)=0.
IF(ISO.EQ.0)GO TO 35
S(I,K)=FAC
GO TO 37
35 DO 38 N=1,LMAX
38 S(I,K)=S(I,K)+.5*FLOAT(2*N-1)*DLSTR(N,M)*PL(N,K)*FAC
37 CONTINUE
36 CONTINUE
DO 813 I=1,NX
SOISTR(I)=0.
DO 813 K=1,NANG
813 SOISTR(I)=SOISTR(I)+S(I,K)*WGT(K)
PRINT 814,(SOISTR(I),I=1,NX)
814 FORMAT(1X,'INITIAL ONCE SCATTERED SOURCE DISTRIBUTION',(10E12.5))
N1=NX-1
SUM=0.5*DX*(SOISTR(1)+SOISTR(NX))
DO 815 I=2,N1
815 SUM=SUM+SOISTR(I)*DX
PRINT 816,SUM
TH=NX*DX
SUM1=ALBEDO(1)*(1.-EXP(TH/ELAMPR(1)))
PRINT 817,SUM1
816 FORMAT(1X,'TRAPEZOIDAL INTEGRATION OF SOURCE FUNCTION',(E12.5))
817 FORMAT(1X,'EXACT INTEGRATION OF SOURCE FUNCTION',(E12.5))
FNORM=SUM1/SUM
DO 818 K=1,NANG
818 S(I,K)=FNORM*S(I,K)
REWIND 7
DO 819 K=1,NANG
819 WRITE(7)(S(I,K),I=1,NX)
40 CONTINUE
C-----FORMAT STATEMENTS-----
10 FORMAT(18I3)
100 FORMAT(6F10.0)
501 FORMAT(1X,'DX=',(E12.5),' CM/CM2 ERR=.',
1F9.6,2X,'ZERO=.',F10.7,2X,'NX=.',13,2X,'NANG=.',13,2X,'NPOS=.',13)
502 FORMAT(1X,'ISOTROPIC SCATTERING')
503 FORMAT(1X,'SCREENED-RUTHERFORD SCATTERING. MATERIAL NO.=',12,
15X,'ETA',(E12.5))
504 FORMAT(1X,'ENERGY OF INCIDENT ELECTRON BEAM = ',F6.3,'MEV./
11X,'GROUP WIDTH = ',F7.4,'MEV.')
505 FORMAT(1X,'COSINE OF INCIDENT BEAM ANGLE = ',E12.5)
506 FORMAT(1X,'MATERIAL NO.=',12/1X,'SLAB THICKNESS = ',E12.5,1X
+,'(CM/CM2)/1X,'SIG(TOTAL CROSS SECTION) = ',E12.5,1X,
+,'(CM2/GM)/1X,'SIG(SCATTERING CROSS SECTION) = ',E12.5,
+1X,'(CM2/GM)/1X,'SIG(ABSORPTION CROSS SECTION) = ',E12.5,
+1X,'(CM/CM2)/1X,'ELAMPR(TOTAL MEAN FREE PATH) = ',E12.5,
+1X,'(CM/CM2)/1X,'ALBEDO = ',F6.3)
508 FORMAT(1X,'GAUSSIAN ORDINATES AT WHICH ALL ANGULAR QUANTITIES ARE
1COMPUTED',(1X,10E12.5))
509 FORMAT(1X,'CORRESPONDING GAUSSIAN WEIGHTS',(1X,10E12.5))
C
E=ENTAB(1)

```



```

NG CROSS
C RETURN
END
SUBROUTINE GROUP
C
COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),ELUX( 51),ESRC
+CUR(12),S( 51,12),SDISTR( 50),CFUX( 50,12),EF( 50),X( 51),
+WT(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NX,NANG,MPOS,
+NSRC,INTRPC,NLAY,E,Z(2),AA(2),IPRNT,COSIN,ENTAB(100),DS(2),
+DISTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,EDGE,ISKP,DELE(50),
+EXCIT(2),STPMR(2),DE,ITER(2),NGRPS,NG,ITMAX,IT,MATL(2),CSUM,
+CMESH(2,10),FBAL(10),IRBL,SUM1,NCRS,STPO,ELAMO
C
DIMENSION ER(20),RBETHE(20,3)
C
C S.D.A. RANGE VALUES IN ALUMINUM, COPPER AND GOLD, RESPECTIVELY,
C ARE GIVEN IN THE ARRAY RBETHE FOR 20 ELECTRON ENERGY VALUES
C CORRESPONDING TO THE ARRAY ER(COMPUTED BELOW). THIS ROUTINE
C CALCULATES ENERGY GROUP WIDTHS CORRESPONDING TO EQUAL PATHLENGTH
C INCREMENTS BY INTEGRATING AN APPROXIMATE EXPRESSION FOR THE
C STOPPING POWER.
C
DATA RBETHE/3.519E-4,7.074E-4,1.165E-3,1.716E-3,2.356E-3,3.08E-3,
13.883E-3,4.762E-3,5.714E-3,6.735E-3,7.822E-3,8.974E-3,1.019E-2,
21.148E-2,1.279E-2,1.417E-2,1.561E-2,1.710E-2,1.864E-2,3.641E-2,
34.558E-4,8.949E-4,1.452E-3,2.121E-3,2.892E-3,3.760E-3,4.719E-3,
45.766E-3,6.896E-3,8.108E-3,9.392E-3,1.075E-2,1.218E-2,1.368E-2,
51.524E-2,1.686E-2,1.858E-2,2.029E-2,2.209E-2,2.427E-2,2.794E-2,
61.453E-3,2.268E-3,3.228E-3,4.319E-3,5.537E-3,6.873E-3,8.322E-3,
79.876E-3,1.153E-2,1.328E-2,1.513E-2,1.706E-2,1.908E-2,2.117E-2,
82.335E-2,2.560E-2,2.792E-2,3.031E-2,3.5749E-2/
MM=MATL(1)
ER(1)=.01
DO 9 I=2,19
9 ER(I)=ER(I-1)*.005
ER(20)=.15
DIFT=10.
DO 10 I=1,20
10 DIF=ABS(ER(I)-E)
IF(DIF.GT.DIFT)GO TO 10
DIFT=DIF
ITAG=I
10 CONTINUE
ECUTE=ECUT(1)*1.03
RB=RBETHE(ITAG,MM)
DS(1)=DS(2)=RB/FLOAT(NGRPS)
IPSAVE=IPRNT
IPRNT=-1
SRANGE=0.
H=0.5*DS(1)
EV=E*1000.
DO 100 J=1,NGRPS
100 CALL RKINT(E,H),RETURNS(20,2)
20 ENTAB(J)=E
NG=J
IF(E.LE.ECUTF)GO TO 2
SRANGE=SRANGE+DS(1)
IF(SRANGE.GE.HB)GO TO 2
100 H=DS(1)
2 NGRPS=NG
IPRNT=IPSAVE
PRINT 3,NGRPS
PRINT 160,ECUTF
3 FORMAT(1X,'NUMBER OF ENERGY GROUPS =',I4)
EOLD=ESRC

```

```

5/ 'GE=0.
6/ JLO
DO 150 J=1,NGRPS
CALL RKINT(E,H),RETURNS(25,5)
25 DELE(J)=EOLD-E
SRANGE=SRANGE+H
IF(SRANGE.GE.RB)GO TO 5
150 EOLD=E
GO TO 85
5 DELE(NGRPS)=EOLD-ECUTF
85 PRINT 161,(J,ENTAB(J),DELE(J),J=1,NGRPS)
160 FORMAT(1X,'CUT OFF ENERGY MEV=',E12.5/13X,'EMID=',5X,'DELE=')
161 FORMAT(1X,15.2E12.5)
RETURN
END
SUBROUTINE RKINT(E,H),RETURNS(M,N)
REAL K1,K2,K3,K4
EOLD=E
IF(E.LE.0.0)RETURN N
K1=STP(E)*H
E=EOLD+.5*K1
IF(E.LE.0.0)RETURN N
K2=STP(E)*H
E=EOLD+.5*K2
IF(E.LE.0.0)RETURN N
K3=STP(E)*H
E=EOLD+K3
IF(E.LE.0.0)RETURN N
K4=STP(E)*H
E=EOLD+1./6.*(K1+2.*K2+2.*K3+K4)
IF(E.LE.0.0)RETURN N
RETURN M
END
SUBROUTINE LEP(Y,X,N)
DIMENSION Y(25)
Y(1)=1.
IF(N)1,1,2
1 RETURN
2 Y(2)=X
IF(N-1)1,1,3
3 DO 4 I=2,N
G=X*Y(I)
4 Y(I+1)=G-Y(I-1)*G-(G-Y(I-1))/FLOAT(I)
RETURN
END
SUBROUTINE CROSS
C
C CALCULATION OF SCREENED RUTHERFORD CROSS SECTION LEGENDRE
C COEFFICIENTS (SEE VOL.1, PAGES 209-214)
C
COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),EFLUX( 51),ESRC
+CUR(12),S( 51,12),SDISTR( 50),CFLUX( 50,12),EF( 50),X( 51),
+NGT(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NX,NANG,NPOS,
+NSRC,INTRFC,NLAY,E,Z(2),AA(2),IPRNT,COSIN,ENTAB(100),DS(2),
+DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,LEDGE,ISKIP,DELE(50),
+EXCIT(2),STPRH(2),DE,ITER(2),NGRPS,NG,ITMAX,IT,MATL(2),CSUM,
+CWESH(2,10),FBAL(10),IRBL,SUMI,NCRS,STPO,ELAMO
C
DIMENSION SGG(25),SIGSTR(25),C(25)
DATA INTIME/0/
DATA TPI/6.2831853/
C-----DEFINITION OF PARAMETERS-----
C
C LMAX = ORDER OF LEGENDRE CROSS SECTION EXPANSION

```



```

IF(DO.GT.DMIN)GO TO 1
DO 1=00
1 CONTINUE
IF(LM.EQ.LMAX)GO TO 2
DO 3 L=L1,LMAX
3 C(L)=Z1
2 CONTINUE
C-----COMPUTE UNCORRECTED SCATTERING CROSS SECTION LEGENDRE COEFFICIENTS
30 SGG(L)=SGG(1)-C(L)*ETA(M)*(1.+5*ETA(M))*SGG(1)
CLAST=C(LMAX)
C-----COMPUTE TRANSPORT CORRECTED SCATTERING CROSS SECTION LEGENDRE
C COEFFICIENTS
DO 31 L=1,LMAX
31 SIGSTR(L)=ETA(M)*(1.+5*ETA(M))*(CLAST-C(L))*SGG(1)
C-----COMPUTE TRANSPORT CORRECTED SCATTERING PROBABILITY LEGENDRE
C COEFFICIENTS
DO 50 L=1,LMAX
50 DLSTR(L,M)=SIGSTR(L)/SIGSTR(1)
C-----ABSORPTION CROSS SECTION
SIGA=1./DS(M)
C-----TRANSPORT CORRECTED TOTAL INTERACTION CROSS SECTION
SIG(M)=(SIGSTR(1)*SIGA)
C-----TRANSPORT CORRECTED MEAN FREE PATH
ELAMPR(M)=1./SIG(M)
C-----TRANSPORT CORRECTED ALBEDO
ALBEDO(M)=SIGSTR(1)/(SIGA+SIGSTR(1))
IF(IPRINT.GE.0)PRINT 590,E=EV,Z(M),STPMR(M),DS(M),ALBEDO(M)
COLL=DS(M)/ELAMPR(M)
IF(IPRINT.LT.0)PRINT 590,E=EV,DS(M),ALBEDO(M),COLL
590 FORMAT(1X,ENERGY = ,F5.5,MEV DS=,E12.5,ALBEDO =,E12.5,
1* COLLISIONS =,E12.5)
IF(IPRINT.LE.0)GO TO 100
C-----PRINT DERIVED QUANTITIES
C
75 LMM=LMAX-1
IF(M.EQ.1)PRINT 502,LMM
76 PRINT 501,M,ETA(M),Z(M),AA(M)
PRINT 503
DO 51 L=1,LMAX
LL=L-1
51 PRINT 504,LL,C(L),SGG(L),SIGSTR(L),DLSTR(L,M)
100 CONTINUE
C-----RESTORE E TO MEV UNITS
E=E*EV
C-----FORMATS
501 FORMAT(1X,MATERIAL NUMBER,12.2X,ETA=,E12.5,
15X,Z(ATOMIC NO.)=,F7.3,5X,AA(ATOMIC WGT.)=,F7.3/)
502 FORMAT(1X,THIS IS AN S=,12.5, CALCULATION)
503 FORMAT(10X,SCREENED RUTHERFORD,5X,TRUE SCATTERING,4X,
+EXTENDED TRANSPORT,3X,TRANSPORT CORRECTED,9X,
+RECURRENCE COEFFICIENTS,4X,CROSS SECTION,9X,
+CORRECTED CROSS,4X,SCATTERING PROBABILITY,5X,LE,12X,
+C(L)=,16X,(CM2/GM)*,8X,SECTION (CM2/GM)*,4X,
+LEGENDRE COEFFICIENTS/)
504 FORMAT(1X,15.5X,E16.9,8X,E16.9,4X,E16.9,8X,E16.9)
510 FORMAT(1X,ISOTROPIC SCATTERING)
590 FORMAT(1X,ENERGY = ,F9.5,MEV, Z =,F7.2,STPMR=,E12.5,
+1X,MEV/(GM/CM2), DS = ,E12.5,GM/CM2,1X,ALBEDO=,E12.5)
C
RETURN
END

```

```

C
SUBROUTINE FLUXES(OUT,JS,NANGP,IS,NXP)
C
      DIMENSION IXP(10),IXM(10)
      DATA INTR/0/
      IF(INTR.GT.0)GO TO 19
      INTR=1
      NXP1=NXP+1
      IX=NX/NCRS
      DO 66 IRR=1,NCRS
        IXP(IRR)=IX*IRR
        IXM(IRR)=1+(IRR-1)*IX
      66 IF(IRBL.GT.0)GO TO 40
      IF(IS.EQ.-1)GO TO 40
      DO 60 IRR=1,NCRS
        DO 60 LL=1,2
      60 CMESH(LL,IRR)=0.
C
C      ZERO OUT NEGATIVE FLUX FIXUP COUNTER FOR THIS ITERATION
C
C      40 NEG=0
C
      IF(JS.NE.1)GO TO 10
      DO 15 I=1,NXP1
      15 EFUX(I)=0.
      DO 100 J=1,NPOS
        IRR=1
        IF(IS.EQ.-1)IRR=NCRS
        JR=J+JS+NANGP
        EF(J)=0.
        IF(IS.EQ.-1)EFUX(NXP1)=EFUX(NXP1)+WGT(J)*EF(J)
        DO 90 I=1,NX
          IF(IS.EQ.-1)GO TO 20
          M=1
          IF(I.GE.INTRFC)M=2
          GO TO 21
      20 M=2
          IF(I.LT.INTRFC)M=1
      21 IR=I+IS+NXP
          IF(IS.EQ.1)EFUX(IR)=EFUX(IR)+WGT(J)*EF(J)
          CFUX(IR,JR)=A(J,M)*EF(J)+B(J,M)*S(IR,JR)
          P=2.*CFUX(IR,JR)
          EF(J)=P-EF(J)
          IF(EF(J).GE.0.)GO TO 30
C
C      NEGATIVE FLUX FIXUP---SET FLUX TO ZERO AND INCREMENT COUNTER
C
      NEG=NEG+1
      EF(J)=0.
      30 CONTINUE
      IF(IRBL.GT.0)GO TO 70
      IF(IS.EQ.-1)GO TO 61
      IF(IR.NE.1)IXP(IRR)GO TO 70
      CMESH(1,IRR)=CMESH(1,IRR)+WGT(J)*AMU(JR)*EF(J)
      IRR=IRR+1
      GO TO 70
      61 IF(IR.NE.1)IXM(IRR)GO TO 70
      CMESH(2,IRR)=CMESH(2,IRR)+WGT(J)*AMU(JR)*EF(J)
      IRR=IRR+1

```



```

+DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,IEDGE,ISKIP,DELE(50)
+E' '(2),STPM(2),DE,ITER(2),NGRPS,IND,ITMAX,IT,MATL(2),CSU
+Q H(2,10),FBAL(10),IRBL,SUM1,NGRS,STPO,ELAMO

      COMPUTE NEW SCATTERING SOURCE

      DO 102 K=1,NANG
      DO 102 I=1,NX
      M=1
      IF(I,GE,INTRFCM-2)
      102 S(I,K)=(1.-ALBEDO(M))*SIG(M)*CFUX(I,K)

      CHECK SOURCE NORMALIZATION INTEGRAL

      DO 114 I=1,NX
      SDISTR(I)=0.
      DO 114 K=1,NANG
      114 SDISTR(I)=SDISTR(I)+S(I,K)*WGT(K)
      110 FORMAT(1X,'SCATTERING SOURCE DISTRIBUTION*',(1X,10E12.5))
      SUM=0.5*DX*(SDISTR(1)+SDISTR(NX))
      NM1=NX-1
      DO 115 I=2,NM1
      DO 115 I=2,NM1
      115 SUM=SUM+SDISTR(I)*DX
      PRINT 111,SUM
      111 FORMAT(1X,'INTEGRAL OF SCATTERING SOURCE = ',E16.8)
      PHORM=(SUM1-CSUM)/SUM
      PRINT 117,PHORM
      117 FORMAT(1X,'SOURCE NORMALIZATION FACTOR=',E12.5)
      REWIND 7
      DO 104 K=1,NANG
      DO 105 I=1,NX
      105 SDISTR(I)=S(I,K)*PHORM
      104 WRITE(7)(SDISTR(I),I=1,NX)
      DO 124 I=1,NX
      SDISTR(I)=0.
      DO 124 K=1,NANG
      124 SDISTR(I)=SDISTR(I)+S(I,K)*WGT(K)
      IF(IPRINT,GT,0)PRINT 110,(SDISTR(I),I=1,NX)
      SUM1=SUM1-CSUM

      COMPUTE NEW SCATTERING CROSS SECTIONS, LEGENDRE COEFFICIENTS

      CALL CROSS

      COMPUTE NEW SCATTERING PROBABILITY

      DO 100 K=1,NANG
      DO 100 L=1,NANG
      DO 100 M=1,NLAY
      PROB=0.
      DO 101 N=1,LMAX
      101 PROB=PROB+.5*FLOAT(2*N-1)*DLSTR(N,M)*PL(N,K)*PL(N,L)
      100 SCAT(K,L,M)=PROB

      COMPUTE CONSTANTS A AND B

      DO 106 J=1,NPOS
      DO 106 M=1,NLAY
      DENOM=2.*AMU(J)*SIG(M)*DX
      A(J,M)=2.*AMU(J)/DENOM
      RETURN
      END
      SUBROUTINE REBAL

```

```

C      JY'S SUBROUTINE PERFORMS THE COARSE MESH REBALANCE PROCEDURE
C      DIMED IN VOL.1, PAGES 199-201
C
C      DIMENSION A1(100),VV(10),FAC(10),CC(10),SS(10),SOLD(50,12)
C      DIMENSION IMARK(10)
C
C      COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),EFLUX( 51),ESRC
C      +CUR(12),S( 51,12),SDISTR( 50),CFUX( 50,12),EF( 50),X( 51),
C      +WGT(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NX,NANG,NCRS,
C      +SRC,INTRFC,NLAY,E,Z(2),AA(2),IPRNT,COSIN,ENTAB(100),DS(2),
C      +DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,IEGGE,ISKIP,DELE(50),
C      +EXCIT(2),STPR(2),DE,ITER(2),NCRPS,NG,ITMAX,IT,MATL(2),CSUM,
C      +CMESH(2,10),FBAL(10),IRBL,SUM1,NCRS,STPO,ELANO
C
C      DATA A1/100*0./
C      DATA FAC/10*1./
C      DATA ITS/0/
C      IF(ITS.GT.0)GO TO 10
C      FAC(1)=FAC(NCRS)=.5
C      ITS=1
C      SA=SIG(1)=1.-ALBEDO(1)
C      IX=NX/NCRS
C      10 DO 11 I=1,NCRS
C      11 CC(I)=0.
C      IF(IT.GT.1)GO TO 16
C      REMIND 7
C      DO 15 J=1,NANG
C      15 READ(7)(SOLD(I,J),I=1,NX)
C      DO 14 I=1,NCRS
C      14 SS(I)=0.
C      DO 12 I=1,IX
C      12 IMARK(I)=J
C      DO 23 NC=2,NCRS
C      23 IMARK(NC)=IMARK(NC-1)+IX
C      DO 22 J=1,NANG
C      FC=WGT(J)*FAC(I)*DX*SA
C      DO 24 NC=1,NCRS
C      24 IM=IMARK(NC)
C      CC(NC)=CC(NC)+FC*CFUX(IM,J)
C      IF(IT.GT.1)GO TO 22
C      IF(NG.EQ.1)GO TO 22
C      FC=WGT(J)*DX*FAC(I)
C      DO 25 NC=1,NCRS
C      25 IM=IMARK(NC)
C      SS(NC)=SS(NC)+FC*SOLD(IM,J)
C      22 CONTINUE
C      12 CONTINUE
C      IF(NG.GT.1)GO TO 17
C      IF(IT.GT.1)GO TO 17
C      X1=0.
C      E1=1.
C      DO 18 I=1,NCRS
C      18 I=IX*DX
C      E2=EXP(-X2)/(ELAMPR(I)*COSIN(I))
C      SS(1)=COSIN*ALBEDO(1)*(E1-E2)
C      X1=X2
C      18 E1=E2
C      17 DO 1 I=1,NCRS
C      17 VV(I)=SS(I)
C      K=(NCRS+1)*(I-1)
C      J=K+1
C      L=K+2
C      A1(J)=CMESH(1,1)-CMESH(2,1)+CC(I)
C      IF(I.LT.NCRS)A1(L)=CMESH(1,1)
C      IF(I.GT.1)A1(K)=CMESH(2,1)
C      1 CONTINUE

```



```

CALL SIMQ(A1,VV,NCRS,KA)
DO 1 I=1,NCRS
4  R(I)=VV(I)
2  IF(KA.GT.0)PRINT 2
  FORMAT(IX,=SOLUTION SUSPECT=)
  RETURN
END
SUBROUTINE SIMQ(A,B,N,KS)
C THIS SUBROUTINE INVERTS A TRIANGULAR MATRIX
C
  DIMENSION A(1),B(1)
  TOL=0.0
  KS=0
  JJ=-N
  DO 65 J=1,N
    JY=J+1
    JJ=JJ+N+1
    BIGA=0
    IT=JJ-J
    DO 30 I=J,N
      IU=IT+1
      IF(ABS(BIGA)-ABS(A(IU)))20,30,30
20  BIGA=A(IU)
      IMAX=I
30  CONTINUE
35  KS=1
      IF(ABS(BIGA)-TOL)35,35,40
      RETURN
40  IT=J+N*(J-2)
      IT=IMAX-J
      DO 50 K=J,N
        I1=I+N
        I2=I1+IT
        SAVE=A(I1)
        A(I1)=A(I2)
        A(I2)=SAVE
        A(I2)=SAVE/BIGA
        SAVE=B(IMAX)
        B(IMAX)=B(J)
        B(J)=SAVE/BIGA
        IF(J=N)55,70,55
55  IQS=N*(J-1)
        DO 65 IX=JY,N
          IXJ=IQS+IX
          IT=J-IX
          DO 60 JX=JY,N
            IXJN=(JX-1)+IX
            JXJ=IXJ+IT
            A(IXJN)=A(IXJN)-(A(IXJ)*A(JXJ))
60  B(IX)=B(IX)-(B(J)*A(IXJ))
70  NY=N-1
            IT=N*N
            DO 80 J=1,NY
              IA=IT-J
              IB=N-J
              IC=N
              DO 80 K=1,J
                B(IB)=B(IB)-A(IA)*B(IC)
              IA=IA-N
              IC=IC-1
80  RETURN
            END
            FUNCTION STP(E)
C STOPPING POWER CALCULATION
C

```

```

C      DA      ITIME/0/
      IF(.TIME.GT.0)GO TO 1
      ITIME=1
      Z=13.
      A=26.98
      EXCIT=(9.76*Z+58.5/(Z**19))*1.E-6
      PRINT 77,EXCIT
      77 FORMAT(1X,'EXCITATION ENERGY=*,E12.5)
      AVOG=6.025E23
      RO=2.8179E-13
      XL2=ALOG(2.)
      FCC=6.2831853*.511*AVOG*RO*RO
C-----CONVERT ENERGY TO MC**2 UNITS
      1 EMEV=E
C
C      USE GROUP MIDPOINT ENERGY
C
C      E=EMEV/.511
C
C      COMPUTE STOPPING POWER IN MEV/(GM/CM2)
      BETA=(SQRT(E*(E+2.)))/(E+1.)
      XLG=ALOG(E*(E+2.))/(2.*(EXCIT/.511)**2))
      FM=1.-BETA*(A*BETA+ (.125*(E-(2.*E+1.))*XL2)/(E+1.))**2
      STP=-((FCC*Z/A/BETA**2)*(XLG+FM)
C
      E=EMEV
      RETURN
      END

```

Program Listing

- UNISORC -

(Reference: Vol. I, Section IV.A.2)


```

C      NANG=2*NPOS
C      R...IND PLOT FILE BUFFER
C      REWIND 8
C      ENERGY GROUP (OUTER ITERATION) LOOP
C      1000 CONTINUE
C      PRINT 511
C-----MAIN (INNER) ITERATION LOOP-----
DO 350 IT=1,ITMAX
  OUT=TRAN*BACK=0.
  CALL FLUXES(TRAN,1,0,1,0)
  CALL FLUXES(BACK,-1,LMAX,-1,NXP)
  OUT=TRAN*BACK
C-----CALCULATE NEW SOURCE-----
  EMAX=0.
  REWIND 7
  DO 200 K=1,NANG
    READ(7) (SDISTR(I),I=1,NX)
    DO 200 I=1,NX
      SUM=SDISTR(I)
      M=1
      IF(1.GE.INTRFC)M=2
      DO 150 L=1,NANG
        SUM=SUM+SCAT(K,L,M)*CFUX(I,L)*WGT(L)*ALBEDO(M)*SIG(M)
        IF(ABS(SUM).LT.ZERO)GO TO 200
        ERROR=ABS((S(I,K)-SUM)/SUM)
        IF(ERROR.GT.EMAX)EMAX=ERROR
      200 S(I,K)=SUM
        IF(IPRINT.LT.1) GO TO 63
        PRINT 512,IT,EMAX,OUT,TRAN,BACK
      63 IF(EMAX.LT.ERR)GO TO 400
      350 CONTINUE
C-----FINAL PRINTOUT FOR THE NG-TH ENERGY GROUP-----
  400 PRINT 505,IT
    IF(IPRINT.LT.2) GO TO 64
C      PRINT CELL CENTER ANGULAR FLUXES
C
    DO 394 I=1,NX
      DO 394 K=LMAX,12
        CFUX(I,K)=0.0
      K1=-7
      395 K1=K1+8
      K2=K1+7
      IF(K2.GT.NANG)K2=NANG
      PRINT 509
      DO 300 I=1,NX,ISKIP
        PRINT 517,I,(CFUX(I,K),K=K1,K2)
        IF(K2.LT.NANG)GO TO 395
      517 FORMAT(1X,14,5X,8E12.5)
      IF(1.EQ.0)GO TO 380
C      COMPUTE EDGE ANGULAR FLUXES(ONLY VALID WHEN DIAMOND DIFF USED)
C      (THE SOURCE ARRAY IS BORROWED HERE FOR THE EDGE FLUX COMPUTATION)
      DO 579 J=1,NANG
        DO 579 J=1,NXP
          S(I,J)=0.0
      579 S(I,J)=0.0
      DO 580 J=1,NPOS
        DO 580 I=2,NXP
          S(I,J)=2.*CFUX(I-1,J)-S(I-1,J)
      580 S(I,J)=2.*CFUX(I-1,J)-S(I-1,J)
      DO 581 J=1,NPOS
        JJ=LMAX-J

```

```

DO 501 I=1,NX
  IF (X(I)-1) 501,502,503
501 S(I,I,0)=2.*CFUX(I,I,0)-S(I,I+1,0)
  K1=-7
396 K1=K1+8
  K2=K1+7
  IF (K2-GT.NANG) K2=NANG
  PRINT 510
  C PRINT CELL EDGE ANGULAR FLUXES
  DO 306 I=1,NXP,ISKIP
306 PRINT 517, I, S(I,K1), K=K1, K2
  IF (K2-LT.NANG) GO TO 396
380 CONTINUE
  84 PRINT 511
  C PRINT CELL EDGE SCALAR FLUXES
  C
  DO 310 I=1,NXP,ISKIP
310 PRINT 516, I, X(I), EFLUX(I)
  C
  C WRITE PLOT FILE
  C
  C WRITE(8)(X(I), EFLUX(I), I=1,NXP)
  C
  C DECREMENT THE ELECTRON ENERGY TO THE NEXT GROUP
  C
  E=E-DE
  NG=NG+1
  IF (NG-GT.NGRPS) GO TO 600
  C
  C COMPUTE NEW CROSS SECTIONS, SCATTERING PROBABILITY, AND
  C SCATTERING SOURCE
  C
  CALL RECALC
  GO TO 1000
  C
  C-----FORMAT STATEMENTS
500 FORMAT(1X,12E11.4)
502 FORMAT(1X,12E11.4)
505 FORMAT(1X,12E11.4)
507 FORMAT(1X,12E11.4)
508 FORMAT(1X,12E11.4)
509 FORMAT(1X,12E11.4)
511 FORMAT(1X,12E11.4)
512 FORMAT(1X,12E11.4)
  C
  C TRANSMISSION=,E12.5, BACKSCATTER=,E12.5,
  C 1-TRANSMISSION=,2X,E13.6/1X, BACKSCATTER=,1X,E13.6)
513 FORMAT(1X,12E11.4)
514 FORMAT(1X,12E11.4)
515 FORMAT(1X,12E11.4)
  C
  C 2X,1, 8X, X(I), 7X, EFLUX(I)
516 FORMAT(1X,13.2X,2E12.5)
518 FORMAT(1X,13.2X,2E12.5)
519 FORMAT(1X,13.2X,2E12.5)
  C
  C (MAXIMUM ALLOWED NUMBER OF INNER ITERATION
  C (NUMBER OF ENERGY GROUPS)/1X,IPRINT=,15, P
  C RINTOUT CONTROL,(0/1/2/3 = FINAL SCALAR FLUX ONLY/ITERATIONS/ANGUL
  C AR FLUX/FULL DEBUG PRINTOUT)/1X,150 *15, (0/1, SCREENED RUTH
  C +ERFORD SCATTERING/ISOTROPIC SCATTERING)/1X,15,
  C * (0/1, ANGULAR EDGE FLUX COMPUTATION, NO/YES)/
  C +1X,15, (FLUX PRINTOUT PARAMETER - PRINT EVERY ISKIP-TH C
  C +ELL)
510 FORMAT(1X,12E11.4)
  C
  C 800 STOP

```

```

END
C  OUTLINE INPUTS
C  COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),EFUX(101)
+CUR(12),S(101,12),SDISTR(100),CFUX(100,12),EF(100),X(101),
+WG(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NA,NANG,NPOS,
+KSRG,INTRFC,MAT,E,Z(2),AA(2),IPRNT,
+DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,IEDGE,ISKIP,
+EXCIT(2),STPWR(2),DE,ITER(2),NGRPS,NG,ITMAX,IT
C  DIMENSION THK(2),Y(25)
C  Y(N)  = OUTPUT ARRAY FROM SUBROUTINE LEP
C  PL(N,K) = LEGENDRE POLYNOMIAL OF ORDER (N-1) FOR THE K-TH
C           GAUSSIAN QUADRATURE ANGLE
C  ZERO OUT DISTRIBUTED SOURCE, CROSS SECTION, ALBEDO
C  AND SCREENING FACTOR ARRAYS
C  DO 101 J=1,100
C    SDISTR(J)=0
C  DO 101 I=1,12
C    101 S(I,J)=0
C  102 SIG(M)=ALBEDO(M)=ETA(M)=0
C  FILL IN THE ATOMIN NUMBER(Z), ATOMIC WEIGHT(AA), AND
C  MEAN EXCITATION ENERGY(EXCIT) ARRAYS.
C  (MATERIAL 1 = ALUMINUM, MATERIAL 2 = GOLD)
C  Z(1)=13
C  Z(2)=79
C  AA(1)=26.98
C  AA(2)=197
C  EXCIT(1)=183.E-8
C  EXCIT(2)=797.E-6
C  NX = NUMBER OF SPATIAL CELLS
C  NPOS = NUMBER OF FIRST QUADRANT QUADRATURE ANGLES
C  INTRFC = SPATIAL INDEX OF MATERIAL BOUNDARY INTERFACE
C  READ 10,NX,NPOS,INTRFC
C  IF THERE IS NO MATERIAL INTERFACE (ONLY ONE SCATTERING
C  MATERIAL), INTRFC IS READ IN AS "0". IT IS THEN SET TO
C  NX+1 (OUTSIDE THE RIGHT BOUNDARY).
C  MAT IS THE NUMBER OF MATERIALS.
C  IF(INTRFC.EQ.0)INTRFC=NX+1
C  MAT=1
C  IF(INTRFC.LT.NX.AND.INTRFC.GT.1)MAT=2
C  DX = SPATIAL CELL WIDTH (CM/CM**2)
C  ERR = TOLERANCE FOR ITERATION CONVERGENCE
C  ZERO = "SAFE" VALUE FOR FLOATING POINT ZERO
C  READ 100,DX,ERR,ZERO
C  E = ELECTRON SOURCE ENERGY (GROUP 1) IN MEV
C  DE = ENERGY GROUP WIDTH IN MEV
C  READ 100,E,DE
C  DEFAULT VALUES FOR DX, ERR, ZERO

```



```

C C C
C IF(DX.EQ.0.)DX=.01
C   ERR.EQ.0.)ERR=.0001
C   ZERO.EQ.0.)ZERO=.00001
C C C
C IF SCATTERING IS ISOTROPIC, READ IN TOTAL CROSS
C SECTION (SIG) AND ALBEDO (ALBEDO) FOR EACH MATERIAL.
C C C
C IF(ISO.GT.0)READ 100,(SIG(M),ALBEDO(M),M=1,MAT)
C NANG=2*NPOS
C PRINT 501,DX,ERR,ZERO,NX,NANG,NPOS
C C C
C COMPUTE GAUSS QUADRATURE ANGLES AND WEIGHTS
C C C
C LMAX=NANG+1
C CALL GRULE(NANG,AMU,WGT)
C DO 15 J=1,NPOS
C   AMU(LMAX-J)=-AMU(J)
C   15 WGT(LMAX-J)=WGT(J)
C C C
C COMPUTE TRANSPORT CORRECTED CROSS SECTIONS (SUBROUTINE
C CROSS) IF SCATTERING IS SCREENED RUTHERFORD.
C OTHERWISE, COMPUTE TOTAL MEAN FREE PATH FOR ISOTROPIC
C SCATTERING.
C C C
C IF(ISO.EQ.0) GO TO 19
C PRINT 502
C DO 18 M=1,MAT
C   ELAMPR(M)=1./SIG(M)
C   GO TO 20
C   19 CALL CROSS
C C C
C COMPUTE THICKNESSES OF MATERIAL ZONES
C C C
C 20 THK(1) = DX*(INTRFC-1)
C IF(MAT.EQ.1)GO TO 2
C THK(1)=DX*(INTRFC-1)
C THK(2)=NX-DX-THK(1)
C DO 14 M=1,MAT
C   SIGS=ALBEDO(M)*SIG(M)
C   SIGA=SIG(M)-SIGS
C   14 PRINT 506,M,THK(M),SIG(M),SIGS,SIGA,ELAMPR(M),ALBEDO(M)
C IF(ISO.GT.0)GO TO 16
C PRINT 503,(M,ETA(M),M=1,MAT)
C PRINT 504,E,DE
C C C
C COMPUTE LEGENDRE POLYNOMIALS
C 18 DO 60 K=1,NANG
C   U=AMU(K)
C   CALL LEP(Y,U,NANG)
C   LMAX=NANG+1
C   DO 60 N=1,LMAX
C     60 PL(N,K)=Y(N)
C C C
C COMPUTE SCATTERING PROBABILITY FUNCTION
C C C
C DO 21 K=1,NANG
C   DO 21 L=1,NANG
C     DO 21 M=1,MAT
C C C
C COMPUTE SCATTERING PROBABILITY
C C C
C PROB=0.5
C ISOTROPIC SCATTERING
C C C
C IF(ISO.GT.0)GO TO 21

```

```

C C SCREENED RUTHERFORD SCATTERING
C
C PROB=0.
DO 22 N=1,LMAX
22 PROB=PROB+0.5*FLOAT(2*N-1)*DLSTR(N,M)*PL(N,K)*PL(N,L)
21 SCAT(K,L,M)=PROB
C
C UNIFORMLY DISTRIBUTED ISOTROPIC SOURCE
C UNITS ARE (CM2/GM--STERADIAN)
C
C REWIND 7
C WIDTH=0.
DO 31 M=1,MAT
31 WIDTH=WIDTH+THK(M)
DO 32 K=1,NANG
DO 33 I=1,NX
33 SDISTR(I)=5/WIDTH
32 WRITE(7)(SDISTR(I),I=1,NX)
40 CONTINUE
C
C-----FORMAT STATEMENTS-----
10 FORMAT(18I3)
100 FORMAT(6F10.0)
501 FORMAT(/IX,DX=,E12.5,GM/CM2 ERR=,
1F9.6,2X,ZERO=,F10.7,2X,NX=,13,2X,NANG=,13,2X,NPOS=,13)
502 FORMAT(/IX,ISOTROPIC SCATTERING=)
503 FORMAT(/IX,SCREENED-RUTHERFORD SCATTERING, MATERIAL NO.,12,
15X,ETA=,E12.5)
504 FORMAT(IX,ENERGY OF INCIDENT ELECTRON BEAM = ,F6.3,MEV/
11X,GROUP WIDTH = ,F7.4,MEV)
506 FORMAT(/IX,MATERIAL NO.,12/IX,SLAB THICKNESS = ,E12.5,1X
+,(GM/CM2)/1X,SIG(TOTAL CROSS SECTION) = ,E12.5,1X,
+,(CM2/GM)*1X,SIG(SCATTERING CROSS SECTION) = ,E12.5,1X,
+,(CM2/GM)/1X,SIG(ABSORPTION CROSS SECTION) = ,E12.5,
+1X,(CM2/GM)/1X,ELAMPR(TOTAL MEAN FREE PATH) = ,E12.5,
+1X,(GM/CM2)/1X,ALBEDO = ,F6.3)
C
C RETURN
C END
C SUBROUTINE LEP(Y,X,N)
C DIMENSION Y(25)
C Y(1)=1.
C IF(N)1,1,2
1 RETURN
2 Y(2)=X
3 IF(N-1)1,1,3
3 DO 4 I=2,N
4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
C RETURN
C END
C SUBROUTINE CROSS
C
C CALCULATION OF SCREENED RUTHERFORD CROSS SECTION LEGENDRE
C COEFFICIENTS(SEE VOL.1, PAGES 209-214)
C
C COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),EFLUX(101)
+CUR(12),S(101,12),SDISTR(100),CFLUX(100,12),EF(100),X(101),
+KRG(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NX,NANG,NPOS,
+KSRFC,INTRFC,MAT,E,Z(2),AA(2),1PRNT
+DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,LEDGE,ISKIP,
+EXCIT(2),STPR(2),DE,ITER(2),NGRPS,NG,ITMAX,IT
C
C DIMENSION SGG(25),SIGSTR(25),C(25),DS(2)

```

```

9 INTIME/0/
C-----DEFINITION OF PARAMETERS-----
C
C LMAX = ORDER OF LEGENDRE CROSS SECTION EXPANSION
C PLUS ONE (MAXIMUM ALLOWED IS ORDER 36)
C
C AVOG = AVOGADRO'S NUMBER
C
C RO = ELECTRON RADIUS(CM.)
C
C SGG(L) = SCATTERING CROSS SECTION LEGENDRE COEFFICIENTS
C
C SIGSTR(L) = TRANSPORT CORRECT SCATTERING CROSS SECTION
C LEGENDRE COEFFICIENTS
C
C DLSTR(L,M) = TRANSPORT CORRECTED SCATTERING PROBABILITY
C
C C(L) = LEGENDRE COEFFICIENTS FOR MATERIAL M
C
C C(L) = SCREENED RUTHERFORD RECURSION RELATION COEFFICIENTS
C USED IN OBTAINING THE LEGENDRE COEFFICIENTS OF THE
C SCATTERING CROSS SECTION
C
C ELAMPR = TRANSPORT CORRECTED MEAN FREE PATH
C
C ALBEDO = TRANSPORT CORRECTED ALBEDO
C-----
C
C IF(INTIME.NE.0)GO TO 95
C AVOG=6.025E23
C RO=2.8179E-13
C XL2=ALOG(2.)
C FCC=6.2831853*.511*AVOG*RO*RO
C INTIME=1
C 95 CONTINUE
C
C-----CONVERT ENERGY TO MC*2 UNITS
C EMEV=E
C
C USE GROUP MIDPOINT ENERGY
C
C E=(EMEV-0.5*DE)/.511
C
C DO 100 M=1,MAT
C LMAX=2*NP05+1
C
C-----COMPUTE SCREENING CONSTANT, ETA(M)
C
C ETA(M)=.5*(Z(M)**(1./3.)/121.245)**2/(E*(2.*E))*(1.13+
C 13.76*(Z(M)/137.)*2*(1.*E)**2/(E*(2.*E)))
C
C-----COMPUTE TOTAL SCATTERING CROSS SECTION
C
C 10 SGG(1)=(1.+Z(M))*AVOG*RO*RO*Z(M)/AA(M)*((1.*E)/(E*(2.*E)))**2
C +/ETA(M)*(1.+5*ETA(M)))
C
C COMPUTE STOPPING POWER IN MEV/(GM/CM2)
C
C BETA=(SORT(E*(E+2.)))/(E+1.)
C XLG=ALOG(E*(E+2.)/(2.*(EXCIT(M)/.511)**2))
C FM=1.-BETA*BETA*(1.125*E*(2.*E+1)*XL2)/(E+1.)*2
C STPWR(M)=(FCC*Z(M)/AA(M)/BETA**2)*(XLG+FM)
C
C COMPUTE ELECTRON PATH LENGTH FOR ENERGY GROUP WIDTH DE
C IN MATERIAL M (DS(M) UNITS ARE GM/CM2)
C
C DS(M)=DE/STPWR(M)
C
C-----COMPUTATION OF SCREENED RUTHERFORD RECURSION COEFFICIENTS.C(L)
C E1=1./((1.+5*ETA(M))
C C(1)=0.

```

```

C/ 1=ALOG(1.+2./ETA(M))-E1
Z 1/ETA(M)
DMIN=1.E6
DO 1 L=3,LMAX
AL=FLOAT(L-2)
BL=2.*AL+1.
CL=AL+1.
C(L)=(BL*(1.+ETA(M))+C(L-1)-CL*C(L-2)-BL*E1)/AL
DO=ABS(C(L)-Z)
IF(DD.GT.DMIN)GO TO 1
DMIN=DO
LM=L
1 CONTINUE
IF(LM.EQ.LMAX)GO TO 2
DO 3 L=L+1,LMAX
3 C(L)=Z1
2 CONTINUE
C-----COMPUTE UNCORRECTED SCATTERING CROSS SECTION LEGENDRE COEFFICIENTS
DO 30 L=2,LMAX
30 SGG(L)=SGG(1)-C(L)*ETA(M)*(1.+5*ETA(M))*SGG(1)
CLAST=C(LMAX)
C-----COMPUTE TRANSPORT CORRECTED SCATTERING CROSS SECTION LEGENDRE
COEFFICIENTS
DO 31 L=1,LMAX
31 SIGSTR(L)=ETA(M)*(1.+5*ETA(M))*(CLAST-C(L))*SGG(1)
C-----COMPUTE TRANSPORT CORRECTED SCATTERING PROBABILITY LEGENDRE
COEFFICIENTS
DO 50 L=1,LMAX
50 DLSTR(L,M)=SIGSTR(L)/SIGSTR(1)
C-----ABSORPTION CROSS SECTION
ABSORB=1./(DS(M)*SIGSTR(1))
SIGA=ABSORB*SGG(1)
C-----TRANSPORT CORRECTED TOTAL INTERACTION CROSS SECTION
SIG(M)=(SIGSTR(1)+SIGA)
C-----TRANSPORT CORRECTED MEAN FREE PATH
ELMPR(M)=1./SIG(M)
C-----TRANSPORT CORRECTED ALBEDO
ALBEDO(M)=SIGSTR(1)/(SIGA+SIGSTR(1))
PRINT 500,EMEV,Z(M),STPMR(M),DS(M)
PRINT 500,EMEV,Z(M),STPMR(M),DS(M)
C-----PRINT DERIVED QUANTITIES
C
75 LMM=LMAX-1
IF(M.EQ.1)PRINT 502,LMM
76 PRINT 501,M,ETA(M),Z(M),AA(M)
IF(IPRINT.LT.2)GO TO 100
PRINT 503
DO 51 L=1,LMAX
LL=L-1
51 PRINT 504,LL,C(L),SGG(L),SIGSTR(L),DLSTR(L,M)
100 CONTINUE
C-----RESTORE E TO MEV UNITS
E=EMEV
C
C-----FORMATS-----
501 FORMAT(1X,'MATERIAL NUMBER',12.2X,'ETA=',E12.5,
15X,'Z(ATOMIC NO.)=',F7.3X,'AA(ATOMIC WGT.)=',F7.3/)
502 FORMAT(1X,'THIS IS AN S-',12,' CALCULATION')
503 FORMAT(10X,'SCREENED RUTHERFORD',5X,'TRUE SCATTERING',4X,
+EXTENDED TRANSPORT,3X,'TRANSPORT CORRECTED',9X,
+RECURRENCE COEFFICIENTS,4X,'CROSS SECTION',6X,
+CORRECTED CROSS',4X,'SCATTERING PROBABILITY',5X,'L',12X,
+*C(L)',16X,'(CM2/GM)',8X,'SECTION (CM2/GM)',4X,
+*LEGENDRE COEFFICIENTS')
504 FORMAT(1X,15.5X,E16.9,8X,E16.9,4X,E16.9,6X,E16.9)

```

```

510 IF 'NAT(IX,0 ISOTROPIC SCATTERING*)
590 K AT(//IX,0 ENERGY = 0.19.5,0 MEV, Z = 0.17.2,0, STPMR = ( 1.5,
+IX,0 MEV/(GM/CM2), DS = 0.12.5,0 GM/CM2*)
C
C RETURN
C END
C SUBROUTINE FLUXES(OUT,JS,NANGP,IS,NXP)
C
C COMMON SCAT(12,12,2),AMU(12),A(12,2),B(12,2),EFLUX(101)
C +,CUR(12),S(101,12),SOISTR(100),CFLUX(100,12),EF(100),X(101),
C +MGT(12),DX,SIG(2),ALBEDO(2),ERR,ZERO,ETA(2),NA,NANG,NPOS,
C +MSRC,INTRFC,MAT,E,Z(2),AA(2),IPRNT,
C +DLSTR(13,2),ELAMPR(2),LMAX,PL(13,12),ISO,EDGE,ISKIP,
C +EXCIT(2),STPMR(2),DE,ITER(2),NGRPS,NG,ITMAX,IT
C
C ZERO OUT NEGATIVE FLUX FIXUP COUNTER FOR THIS ITERATION
C
C NEG=0
C
C IF(JS.NE.1)GO TO 10
C NXP1=NXP+1
C DO 15 I=1,NXP1
C 15 EFLUX(I)=0.
C 10 DO 100 J=1,NPOS
C JR=J+JS+NANGP
C EF(J)=0.
C IF(15.EQ.-1)EFLUX(NXP1)=EFLUX(NXP1)+MGT(J)*EF(J)
C DO 90 I=1,NX
C IF(15.EQ.-1)GO TO 20
C M=1
C IF(1,GE,INTRFC)M=2
C GO TO 21
C 20 M=2
C IF(1,LT,INTRFC)M=1
C 21 IR=15+NXP
C IF(15.EQ.1)EFLUX(IR)=EFLUX(IR)+MGT(J)*EF(J)
C CFLUX(IR,JR)=A(J,M)*EF(J)+B(J,M)*S(IR,JR)
C P=2.*CFLUX(IR,JR)
C EF(J)=P-EF(J)
C IF(EF(J).GE.0.)GO TO 30
C
C NEGATIVE FLUX FIXUP---SET FLUX TO ZERO AND INCREMENT COUNTER
C
C NEG=NEG+1
C EF(J)=0.
C 30 CONTINUE
C IF(15.EQ.-1)EFLUX(IR)=EFLUX(IR)+MGT(J)*EF(J)
C 90 CONTINUE
C IF(15.EQ.1)EFLUX(NXP1)=EFLUX(NXP1)+MGT(J)*EF(J)
C C-----DETERMINE TOTAL OUTGOING CURRENT-----
C CUR(JR)=EF(J)*AMU(J)
C OUT=OUT+CUR(JR)*MGT(JR)
C 100 CONTINUE
C
C NEGATIVE FLUX FIXUP PRINTOUT
C
C IF(NEG.GT.0)PRINT 150,IT,NEG
C 150 FORMAT(1X,0 ITERATION NO.,0,15,5X,110,0 NEGATIVE FLUX FIXUPS*)
C RETURN
C END
C SUBROUTINE GRULE(N,X,M)
C -----GENERATE GAUSS QUADRATURE ABSCISSAS AND WEIGHTS---
C DIMENSION X(1),W(1)
C M=(N+1)/2
C ET=N*(N+1)

```


Program Listing

- MULTSCT -

(Reference: Vol. I, Section IV.B.3)


```

PROGRAM MULTSCT(INPUT,OUTPUT,TAPE2,TAPE8)

C
C
C PROGRAM MULTSCT IS A CONDENSED COLLISION MONTE CARLO PROGRAM FOR
C SOLVING THE ELECTRON TRANSPORT PROBLEM OF AN ELECTRON BEAM
C INCIDENT ON A MATERIAL SLAB. THE ALGORITHM IS BASED ON THE METHOD
C OF CURGUVEN AND DUMCUMB, WHICH WAS FURTHER DEVELOPED BY LOVE
C AND MYKALEBUST (SEE REFS.15-17, P.265, VOL. I). THE RANGE CORRE-
C SPONDING TO THE ENERGY OF THE INCIDENT ELECTRON IS DIVIDED INTO
C 100 EQUAL TRAJECTORY STEPS. THE ELECTRON ENERGY AT THE MIDPOINT
C OF EACH STEP IS COMPUTED IN SUBROUTINE "ESET" BY RUNGE-KUTTA INTE-
C GRATION OF THE STOPPING POWER FORMULA(EQ.40, P.231, VOL.1)

C
C
C COMMON/MC/NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
C 1EO,N,IFLAG,MAXCOLL,LCHR,TLIM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
C 2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRNT,TLAST,NBATCH,ENTAB(100),INCODE(3),
C 3XIRC,MAT.ONSTY(3),STO(4,101),TBO,DRB,RTOTAL,BCHF(3),NRA,MARA,FORCE
C 4,THETA
C
C DIMENSION TIMR(11)

C
C SUBROUTINE "SETRUN" READS IN AND INITIALIZES THE VARIOUS PARAMETERS
C NECESSARY FOR THE MONTE CARLO CALCULATION

C
C CALL SETRUN
C FN=FLOAT(NMAX)
C COLLNO=0.
C MTM=1
C
C CALL SECOND(TIME)
C TIMR(1)=TIME
C
C 5 N=N+1

C
C "SETHS" INITIALIZES THE ELECTRON TRAJECTORY PARAMETERS AND IS
C CALLED AT THE START OF EACH TRAJECTORY.

C
C CALL SETHS
C "SCORE" RECORDS THE COORDINATES, DIRECTION AND ENERGY AT EACH
C ELECTRON COLLISION SITE

C
C CALL SCORE

C
C "PENET" TRANSLATES THE ELECTRON FROM ONE COLLISION SITE TO THE
C NEXT

C
C 20 CALL PENET
C IF(IFLAG.NE.0) GO TO 6

C
C "ENERGY" ASSIGNS THE ELECTRON ENERGY AFTER A COLLISION HAS
C OCCURRED

C
C CALL ENERGY
C IF(IFLAG.NE.0) GO TO 6
C CALL SCORE
C CALL ANGLES
C IF(IFLAG.NE.0)GO TO 6
C GO TO 20

C
C 6 CALL SCORE
C NRAP=NRAP+1
C DO 7 MCC=NRAP,MARA
C RA=РАНF(0)
C RA=РАНF(0)
C RA=РАНF(0)
C
C 7 CONTINUE

```



```

C      R( IN AND INITIALIZE ALL PARAMETERS NECESSARY FOR THE (
C      CALCULATION.
C
COMMON/MC/MACCOLL,CTHO,CTH,STM,PHI,CPHI,EGY,CPHIO,SPHIO,DELTE,
1EQ,N,IFLAG,MACCOLL,LCHR,TLM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRNT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TRO,DBR,RTOTAL,BCKF(3),NRA,NKRA,FORCE
4,THETA
DATA CPHIO,SPHIO,ZMIN,ZMAX,N/1.0,0.0,0.0,1000.0/
DATA XO,YO,ZO/3.0,0./
DATA DNSTY/1.18,96.19,3/
DATA BCKF/1.1,260.417/
READ 1,NMAX,MAT,NPRNT,(INCODE(1),I=1,3)
MACCOLL=100
C
C      NMAX - TOTAL NO. OF MONTE CARLO HISTORIES
C      MACCOLL - MAXIMUM NUMBER OF COLLISIONS PER HISTORY TO BE ALLOWED
C      NPRNT - DEBUG PRINT CONTROL PARAMETER
C      NPRNT EQUAL TO ZERO SUPPRESSES DEBUG PRINTOUT
C      INCODE- INPUT KEYS FOR BULK BACKSCATTER FRACTION, CHANGES IN
C      SLAB THICKNESSES, RANDOM NUMBER INITIALIZATION
C      TLM - TIME LIMIT(SEC) FOR CALCULATION
C      CTHO - INITIAL POLAR ANGLE OBLIQUITY COSINE OF BEAM
C      ED - ELECTRON BEAM ENERGY(KEV)
C      ECUT - ELECTRON CUT-OFF ENERGY
C
READ 3,TLM,CTHO,ED,ECUT
ED=.001* J
ECUT=.001*ECUT
IF(INCODE(1).GT.0)READ 3,BCKF(MAT)
IF(INCODE(1).GT.0)PRINT 30,BCKF(MAT)
30 FORMAT(1X,'BULK BACKSCATTER FRACTION =',F7.4)
CALL ESET
DTH=1.5E-5
C
C      FORCE - APPLIED ELECTRIC FIELD (EV/CM)
C
READ 3,FORCE
PRINT 35,FORCE
FORCE=FORCE/DNSTY(MAT)/1.E6
PRINT 36,FORCE
35 FORMAT(1X,'FIELD FORCE (EV/CM)=',E12.5)
36 FORMAT(1X,'FIELD FORCE(MEV PER G/CM2)=',E12.5)
DO 50 I=1,20
50 THICK(I)=I*OTH
20 FORMAT(8F10.0)
THICK(21)=3.15E-4
IF(INCODE(2).GT.0)READ 20,(THICK(I),I=1,21)
ZMAX=THICK(21)
INRAN=0
IF(INCODE(3).GT.0)READ 8,INRAN
EO - INITIAL PARTICLE ENERGY
XO,YO,ZO - INITIAL PARTICLE COORDINATES
PRINT 4,NMAX,MACCOLL
IF(MAT.EQ.1)PRINT 15
IF(MAT.EQ.2)PRINT 16
IF(MAT.EQ.3)PRINT 17
C      USUALLY "ALUMINUM" IS ASSUMED INSTEAD OF "LUCITE" HERE
C
15 FORMAT(1X,'LUCITE+')
16 FORMAT(1X,'COPPER+')
17 FORMAT(1X,'GOLD+')
9 FORMAT(1X,'ED (INITIAL ENERGY MEV)=',E12.5//1X,'ECUT (LOW ENERGY

```

```

1C1" OFF MEV) *E12.5//1X,*XO, YO, ZO, (SOURCE POINT COORDINATES)
3 (.3E12.5//1X,*CTHO (INCIDENT POLAR COSINE) *E12.5// CPHIO
4, S-HID (INCIDENT AZIMUTH) *2E12.5)
8 FORMAT(018)
1 FORMAT(615)
3 FORMAT(8F10.0)
4 FORMAT(//1X,*NMAX (NO. OF HISTORIES) *16//1X,*MAXCOLL (MAXIMUM
1 ALLOWED NUMBER OF COLLISIONS) *,13)
10 N=0
XIRA=XIRC=INRAN
MXRA=MAXCOLL+10
NBATCH=NMAX/10
CALL RANSET(XIRA)
PRINT 9,EO,ECUT,XO,ZO,CTHO,CPHIO,SPHIO
RETURN
END
SUBROUTINE SETHIS
C
C INITIALIZE ELECTRON HISTORY
COMMON/MC/MCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EO,N,IFLAG,MAXCOLL,LCHR,TLIM,XO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DRB,RTOTAL,BCKF(3),NRA,MARA,FORCE
4,THETA
LCHR=0
NCOLL=0
NRA=0
X=XO
Y=Y0
Z=ZO
CPHI=CPHIO
SPHI=SPHIO
PHI=0.0
CTH=CTHO=1.
THETA=0.
STH=SQRT(1.-CTH**2)
EGY=EO
IFLAG=0
DELTE=0.
RETURN
END
SUBROUTINE PENET
C
C CALCULATE COORDINATES OF NEXT COLLISION POINT AT DISTANCE "DRB"
C AWAY FROM PREVIOUS COLLISION POINT
C
COMMON/MC/MCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EO,N,IFLAG,MAXCOLL,LCHR,TLIM,XO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DRB,RTOTAL,BCKF(3),NRA,MARA,FORCE
4,THETA
S=DRB
X1=X
Y1=Y
Z1=Z
DELZ=.25/EGY*FORCE=DRB*DRB
Z-Z1=S*CTH*DELZ
IF(Z.GT.ZMIN) GO TO 10
Z=ZMIN
S=(ZMIN-Z1)/CTH
IFLAG=1
10 IF(Z.LT.ZMAX) GO TO 11
Z=ZMAX
S=(ZMAX-Z1)/CTH
IFLAG=1

```

```

11 X1+S+STM+CPHI
    +S+STM+SPHI
    NCOLL=NCOLL+1
    IF(NCOLL.EQ.MAXCOLL) IFLAG=1
    RETURN
END
SUBROUTINE ESET
C CALCULATE ELECTRON ENERGIES CORRESPONDING TO 100 EQUAL
C TRAJECTORY STEPS
C
COMMON/MC/NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EO,N,IFLAG,MAXCOLL,LCHR,TLM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DBR,RTOTAL,BCKF(3),NRA,MXRA,FORCE
4,THETA
DIMENSION ZM(3),AM(3)
DIMENSION ER(20),RBETHE(20,3)
C RBETHE IS A TABLE OF ELECTRON C.S.D.A. RANGES FOR 20 ENERGIES
C CORRESPONDING TO THE COMPUTED ARRAY "ER" BELOW, FOR ALUMINUM,
C COPPER AND GOLD. THIS TABLE IS NOT ALWAYS USED. IN THE PRESENT
C CALCULATION, THE RANGE FOR LUCITE(200 KEV) IS ENTERED VIA
C REPLACEMENT STATEMENT (SEE BELOW .. "RTOTAL =...")
C
DATA RBETHE/3.519E-4,7.074E-4,1.165E-3,1.716E-3,2.356E-3,3.08E-3,
13.883E-3,4.762E-2,5.714E-2,3.673E-3,7.822E-3,6.974E-3,1.019E-2,
21.146E-2,1.279E-2,1.417E-2,1.561E-2,1.710E-2,1.864E-2,3.641E-2,
34.556E-4,8.949E-4,1.453E-3,2.121E-3,2.892E-3,3.760E-3,4.719E-3,
45.766E-3,6.896E-3,8.106E-3,9.392E-3,1.075E-2,1.218E-2,1.368E-2,
51.524E-2,1.686E-2,1.855E-2,2.029E-2,2.209E-2,2.477E-2,7.944E-4,
61.453E-3,2.268E-3,3.226E-3,4.319E-3,5.537E-3,6.873E-3,8.322E-3,
79.876E-3,1.153E-2,1.328E-2,1.513E-2,1.706E-2,1.908E-2,2.117E-2,
82.335E-2,2.580E-2,2.792E-2,3.031E-2,3.274E-2,5.748E-2/
DATA ZM,AM/5.901,25.,79.,10.785,63.54,197./
FD(X,CY)=CX/X*(ALOG(1.166*X)-CY)
ZA=ZM(MAT)
ER(1)=.01
DO 250 I=2,20
    ER(I)=ER(I-1)+.005
250 ER(I)=ER(I-1)+.005
BF=BCKF(MAT)
TBO=.02209+.10718*BF+.03009*BF*BF+.37555*BF*BF*BF
AA=AM(MAT)
RTOTAL=.27866E-3
DRB=RTOTAL/100.
I=1
H=RB=.5*DRB
EV=ED*1000.
C1=-78500.*ZA/AA
C2=ALOG(.001*(9.76*ZA*58.5/((ZA**19)))
EOLD=EV
11 CAY1=FD(EV,C1,C2)*H
EV=EOLD+.5*CAY1
CAY2=FD(EV,C1,C2)*H
EV=EOLD+.5*CAY2
CAY3=FD(EV,C1,C2)*H
EV=EOLD+CAY3
CAY4=FD(EV,C1,C2)*H
ENTAB(1)=EOLD+1./6.*(CAY1+2.*CAY2+2.*CAY3+CAY4)
EOLD=EV+ENTAB(1)
IF(1.EQ.100) GO TO 105
I=I+1
H=2.*RB
GO TO 11
105 PRINT 52
H=RB

```

```

C 50 I=1,100
C IF 51-H,ENTAB(1)
50 H=1-2-RR
51 FORMAT(1X,E12.5,10X,E12.5)
52 FORMAT(6X,=RANGE=,7X,=MIDPOINT ENERGY(KEV)=)
DO 53 I=1,100
53 ENTAB(I)=.001*ENTAB(1)
IF(ECUT.GT.ENTAB(100))RETURN
ECUT=ENTAB(100)
RETURN
END
SUBROUTINE ANGLES
C COMPUTE POLAR AND AZIMUTHAL ANGLES OF ELECTRON TRAJECTORY AFTER
C COLLISION
C
COMMON/MC/NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EO,N,IFLAG,MAXCOLL,LCHR,TLIM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DBR,RTOTAL,BCKF(3),NRA,NXRA,FORCE
4,THETA
DATA TWOPI/6.283185307/
DTH=ABS(2.*DELZ/DRB)
IF(FORCE.GT.0.)THETA=THETA-DTH
IF(THETA.LT.0.)THETA=-THETA
IF(FORCE.LT.0.)THETA=THETA+DTH
CTH=COS(THETA)
STH=SQRT(1.-CTH*CTH)
RD=RANF(0)
TBO=(TBO*EQ/EGY)*.2/RD
COM=(1.-TBO)/(1.+TBO)
SOM=SQRT(1.-COM*COM)
6 RC=RANF(0)
NRA=NRA+1
RHO=TWOPI*RC
CRHO=COS(RHO)
SRHO=SIN(RHO)
STH1=STH
CTH1=CTH
CTH=CTH1*COM+STH1*SOM*CRHO
STH=SQRT(1.-CTH*CTH)
IF(STH1.EQ.0.0.OR.STH.EQ.0.0) GO TO 1
C1=(COM-CTH*CTH1)/(STH*STH1)
S1=SRHO*SOM/STH
CPHI1=CPHI
SPHI1=SPHI
CPHI=CPHI1+C1*SPHI1*S1
SPHI=CPHI1+S1*SPHI1*C1
GO TO 2
1 CPHI=CRHO
SPHI=SRHO
2 CONTINUE
PHI=ATAN2(SPHI,CPHI)
IF(PHI.LT.0)PHI=PHI+TWOPI
THETA=ACOS(CTH)
THETA=ACOS(CTH)
RETURN
END
SUBROUTINE SCORE
C RECORD ELECTRON TRAJECTORY CHARACTERISTICS
C
COMMON/MC/NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EO,N,IFLAG,MAXCOLL,LCHR,TLIM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DBR,RTOTAL,BCKF(3),NRA,NXRA,FORCE

```



```

NY 4+1
NA I=NHIST+1
IOLD=1
I2TOP=1
Z1=0.
GO TO 101
5 Z=STO(1,L)
NCOLL=IFIX(ABS(STO(3,L))*.1)
EGY=STO(4,L)
MU=STO(2,L)
IF(MU)70,70,80
70 DO 71 I=1,19
71 IF(Z.LT.THICK(I).AND.Z1.GT.THICK(I))EFLNEG(I+1)=EFLNEG(I)+
11000.*EGY
GO TO 85
80 DO 81 I=1,20
81 IF(Z.GT.THICK(I).AND.Z1.LT.THICK(I))EFLPOS(I)=EFLPOS(I)+
11000.*EGY
85 Z1=Z
AMU=ABS(MU)
ARG=2.*AMU-1.
C
C EVALUATE LEGENDRE POLYNOMIALS
C
C CALL LEP(PL,ARG,5)
C
C CLASSIFY EMERGENT ENERGY
DO 25 K=1,10
25 IF(EGY.GT.ENERGY(K))GO TO 26
K=10
26 KC=K
EF=EGY/EO
KK=IFIX(20.*EF)+1
FA(I)=FB(I)=1.
DO 111 MA=2,5
FA(MA)=AMU*FA(MA-1)
111 FB(MA)=1000.*EGY*FB(MA-1)
C
C
C IF(Z.GT.ZMIN)GO TO 6
C IF(I2TOP.GT.20)I2TOP=20
C BACKSCATTERED CURRENT
DO 11 I=I2TOP,20
BC(I,KC)=BC(I,KC)+1.0
BTOT(I)=BTOT(I)+1.
C
C LEGENDRE COEFFICIENTS OF BACKSCATTERED ANGULAR DISTRIBUTION
DO 1101 LP=2,6
BPL(I,KC,LP-1)=BPL(I,KC,LP-1)+PL(LP)
1101 BPLTOT(I,LP-1)=BPLTOT(I,LP-1)+PL(LP)
11 CONTINUE
BFE(KK)=BFE(KK)+1.0
C
C BACKSCATTER ENERGY-ANGLE MOMENTS MATRIX
DO 1102 MA=1,5
DO 1102 MB=1,5
1102 BROW(MA,MB)=BROW(MA,MB)+FA(MA)*FB(MB)
EFLNEG(1)=EFLNEG(1)+1000.*EGY
GO TO 102
6 DO 10 IZ=1,21

```



```

IF(Z.GT.THICK(I2))GO TO 10
I( 2,IG.1)GO TO 102
I( 2,IG.1)GO TO 102
IF(15.LE.IZTOP)GO TO 102
IZTOP=15
IQ=15-1
C
C TRANSMITTED CURRENT
C
DO 12 IT=IOLD,IQ
TC(IT,KC)=TC(IT,KC)+1.
TTOT(IT)=TTOT(IT)+1.
TFE(KK,IT)=TFE(KK,IT)+1.0
C
C TRANSMISSION ENERGY-ANGLE MOMENTS MATRIX
C
DO 1103 MA=1,5
DO 1103 MB=1,5
1103 TMOM(MA,MB,IT)=TMOM(MA,MB,IT)+FA(MA)*FB(MB)
C
C LEGENDRE COEFFICIENTS OF TRANSMISSION ANGULAR DISTRIBUTION
C
DO 1201 LP=2,6
TPL(IT,KC,LP-1)=TPL(IT,KC,LP-1)+PL(LP)
1201 TPLTOT(IT,LP-1)=TPLTOT(IT,LP-1)+PL(LP)
12 CONTINUE
IOLD=IZTOP
GO TO 102
10 CONTINUE
102 IF(STO(3,L).LT.0.0)GO TO 112
GO TO 101
112 IF(NH.LT.NNN)GO TO 101
KPUT=1
NH=0
1B=1B+1
WRITE(6)1B
C
C ENERGY FLOW CALCULATION FOR 20 ZONES
C
C *EFLNET* IS THE NET ENERGY DEPOSITION
C
EFLNET(1)=1000.*FN*EO-EFLPOS(1)+EFLNEG(2)-EFLNEG(1)
EFLNET(20)=-EFLPOS(20)-EFLNEG(20)+EFLPOS(19)
DO 86 I=2,19
86 EFLNET(I)=EFLNEG(I+1)-(EFLNEG(I)+EFLPOS(I))+EFLPOS(I-1)
DO 103 I=1,20
TTOT(I)=TTOT(I)/FN
BTOT(I)=BTOT(I)/FN
EFLPOS(I)=EFLPOS(I)/FN
EFLNEG(I)=EFLNEG(I)/FN
EFLNET(I)=EFLNET(I)/FN
C
C THE REMAINING SECTION OF CODE CALCULATES THE STATISTICAL
C STANDARD ERROR IN THE COMPUTED OUTPUT QUANTITIES. WHEN ALL
C HISTORIES HAVE BEEN PROCESSED, THE RESULTS ARE PRINTED OUT.
C
DO 1104 MA=1,5
DO 1104 MB=1,5
1104 TMOM(MA,MB,1)=TMOM(MA,MB,1)/FN
DO 103 LP=1,5
BPLTOT(I,LP)=BPLTOT(I,LP)/FN
TPLTOT(I,LP)=TPLTOT(I,LP)/FN
DO 103 KK=1,10
BPL(I,K,LP)=BPL(I,K,LP)/FN
TPL(I,K,LP)=TPL(I,K,LP)/FN
103 CONTINUE
DO 113 I=1,20

```

```

DO 113 K=1,10
TC(I,K)=TC(I,K)/FN
113 IF (I.EQ.1) BC(I,K)=BC(I,K)/FN
DO 723 KK=1,20
BFE(KK)=BFE(KK)/FN
DO 723 I=1,20
TFC(KK,I)=TFC(KK,I)/FN
DO 1105 MA=1,5
DO 1105 MB=1,5
1105 BMON(MA,MB)=BMON(MA,MB)/FN
IF (I.EQ.10) CALL SECOND(TLAST)
DO 150 NM=1,2
DO 104 K=1,10
DO 105 I=1,20
GO TO (115,125),NM
115 TEMP(I)=TC(I,K)
TFC(I,K)=0.
GO TO 105
125 TEMP(I)=BC(I,K)
BC(I,K)=0.
105 CONTINUE
C
C SUBROUTINE 'STATS' STORES THE OUTPUT QUANTITIES AND COMPUTES
C THEIR STANDARD ERRORS
C
CALL STATS(TEMP,ISIG,20,KPUT,18)
WRITE(6)(TEMP(I),ISIG(I),I=1,20)
IF (I.EQ.10) GO TO 104
DO 156 I=1,20
IF (NM.EQ.1) TFC(I,K)=TEMP(I)
IF (NM.EQ.2) BC(I,K)=TEMP(I)
156 IG(I,K)=ISIG(I)
104 CONTINUE
DO 140 I=1,20
GO TO (141,142),NM
141 TEMP(I)=TTOT(I)
TTOT(I)=0.
GO TO 140
142 TEMP(I)=BTOT(I)
BTOT(I)=0.
140 CONTINUE
CALL STATS(TEMP,ISIG,20,KPUT,18)
WRITE(6)(TEMP(I),ISIG(I),I=1,20)
IF (I.EQ.10) GO TO 150
DO 143 I=1,20
IF (NM.EQ.1) TTOT(I)=TEMP(I)
IF (NM.EQ.2) BTOT(I)=TEMP(I)
143 IGOT(I)=ISIG(I)
IF (I.EQ.10) GO TO 150
PRINT 220
DO 1531 KK=1,2
I1=IL(KK)
I2=IH(KK)
GO TO (114,124),NM
114 PRINT 230
GO TO 153
124 PRINT 231
153 PRINT 236, (THICK(I),I=1,12)
236 PRINT 236, (TNM(I),I=1,12)
230 FORMAT(5X,'NM=',2X,10E12.5/3X,'E(KEV)=')
230 FORMAT(//53X,'TRANSMITTED CURRENT=')
231 FORMAT(//53X,'BACKSCATTERED CURRENT=')
234 FORMAT(53X,'SLAB THICKNESS/2X',(G/CM2),10E12.5)
DO 158 K=1,10
GO TO (1581,1582),NM
1581 PRINT 221,ENERG(K), (TC(I,K),IG(I,K),I=1,12)

```

```

WRITE(2)ENERK(K),(TC(I,K),IG(I,K),I=1,12)
C O 158
1582 PA,NT 221,ENERK(K),(BC(I,K),IG(I,K),I=1,12)
WRITE(2)ENERK(K),(BC(I,K),IG(I,K),I=1,12)
158 CONTINUE
GO TO(1583,1584),MM
1583 PRINT 235,(TTOT(I),IGTOT(I),I=1,12)
WRITE(2)((TTOT(I),IGTOT(I),I=1,12)
GO TO 1531
1584 PRINT 235,(BTOT(I),IGTOT(I),I=1,12)
WRITE(2)((BTOT(I),IGTOT(I),I=1,12)
235 FORMAT(/2X,'TOTALS',1X,10(F9.5,13))
1531 CONTINUE
150 CONTINUE
DO 175 MM=1,3
DO 180 I=1,20
GO TO (181,182,183),MM
181 TEMP(I)=EFLNEG(I)
EFLNEG(I)=0.
GO TO 180
182 TEMP(I)=EFLPOS(I)
EFLPOS(I)=0.
GO TO 180
183 TEMP(I)=EFLNET(I)
EFLNET(I)=0.
180 CONTINUE
CALL STAYS(TEMP,ISIG,20,KPUT,18)
WRITE(6),TEMP(I),ISIG(I),I=1,20)
IF(18-LT,10)GO TO 175
DO 184 I=1,20
IF(MM.EQ.1)EFLNEG(I)=TEMP(I)
IF(MM.EQ.2)EFLPOS(I)=TEMP(I)
IF(MM.EQ.3)EFLNET(I)=TEMP(I)
184 IGTOT(I)=ISIG(I)
175 CONTINUE
IF(18-LT,10)GO TO 176
WRITE(2)(EFLNET(I),I=1,20)
ZERO=0.
I2=1
PRINT 185
PRINT 186,I2,ZERO,EFLNEG(1),THICK(1),EFLPOS(1),EFLNET(1)
DO 187 I=2,20
187 PRINT 186,I,THICK(I-1),EFLNEG(I),THICK(I),EFLPOS(I),EFLNET(I)
185 FORMAT(/30X,'ENERGY DEPOSITION PROFILE',10X,'ZONE',6X,'Z',10X,
1-EFLNEG',8X,'Z',8X,'EFLPOS',5X,'EFLNET')
186 FORMAT(12X,12,1X,E12.5,1X,F10.7,E12.5,1X,F10.7,1X,F10.7)
176 DO 350 MM=1,2
DO 300 LP=1,5
DO 300 K=1,10
DO 301 I=1,20
GO TO (302,312),MM
302 TEMP(I)=TPL(I,K,LP)
TPL(I,K,LP)=0.
GO TO 301
312 TEMP(I)=BPL(I,K,LP)
BPL(I,K,LP)=0.
301 CONTINUE
CALL STAYS(TEMP,ISIG,20,KPUT,18)
WRITE(6)(TEMP(I),ISIG(I),I=1,20)
IF(18-LT,10)GO TO 300
DO 303 I=1,20
TPL(I,K,LP)=TEMP(I)
303 IPL(I,K,LP)=ISIG(I)
300 CONTINUE
DO 320 LP=1,5
DO 321 I=1,20

```

```

322      GO TO (322,332),MM
      TPLTOT(I,LP)=0.
      GO TO 321
322  TEMP(I)=BPLTOT(I,LP)
      BPLTOT(I,LP)=0.
321  CONTINUE
      CALL STATS(TEMP,ISIG,20,KPUT,18)
      WRITE(6)(TEMP(I),ISIG(I),I=1,20)
      IF(18.LT.10)GO TO 320
      DO 323 I=1,20
      TPLTOT(I,LP)=TEMP(I)
323  IG(I,LP)=ISIG(I)
320  CONTINUE
      IF(18.LT.10)GO TO 350
      PRINT 220
      GO TO (314,324),MM
314  PRINT 340
      GO TO 325
324  PRINT 341
325  E1=1000.*ED
      DO 326 K=1,10
      E2=ENERG(K)
      PRINT 342,E1,E2
      E1=E2
      PRINT 343
      DO 327 I=1,20
      GO TO(3271,3272),MM
3271  WRITE(2)TC(I,K),(TPL(I,K,LP),IPL(I,K,LP),LP=1,5)
      IF(TC(I,K).NE.0.0)PRINT 344,THICK(I),TC(I,K),(TPL(I,K,LP),
      IPL(I,K,LP),LP=1,5)
      GO TO 327
3272  WRITE(2)BC(I,K),(TPL(I,K,LP),IPL(I,K,LP),LP=1,5)
      IF(BC(I,K).NE.0.0)PRINT 344,THICK(I),BC(I,K),(TPL(I,K,LP),
      IPL(I,K,LP),LP=1,5)
327  CONTINUE
326  CONTINUE
      PRINT 345
      PRINT 343
      DO 328 I=1,20
      GO TO (3281,3282),MM
3281  WRITE(2)TTOT(I),(TPLTOT(I,LP),IG(I,LP),LP=1,5)
      IF(TTOT(I).NE.0.0)PRINT 344,THICK(I),TTOT(I),(TPLTOT(I,LP),
      IG(I,LP),LP=1,5)
      GO TO 328
3282  WRITE(2)BTOT(I),(TPLTOT(I,LP),IG(I,LP),LP=1,5)
      IF(BTOT(I).NE.0.0)PRINT 344,THICK(I),BTOT(I),(TPLTOT(I,LP),
      IG(I,LP),LP=1,5)
328  CONTINUE
350  CONTINUE
      DO 725 I=1,20
      DO 726 K=1,20
      TEMP(K)=TFE(K,I)
      TFE(K,I)=0.
726  CALL STATS(TEMP,ISIG,20,KPUT,18)
      WRITE(6)(TEMP(K),ISIG(K),K=1,20)
      IF(18.LT.10)GO TO 725
      DO 727 K=1,20
      TFE(K,I)=TEMP(K)
727  IFE(K,I)=ISIG(K)
725  CONTINUE
      DO 728 K=1,20
      TEMP(K)=BFE(K)
      BFE(K)=0.
728  CALL STATS(TEMP,ISIG,20,KPUT,18)
      WRITE(6)(TEMP(I),ISIG(I),I=1,20)

```

```

17 8.LT.10)GO TO 770
DO 30 K=1,20
  BFE(K)=TEMP(1)
730 IBFE(K)=ISIG(K)
770 DO 775 MA=1,5
  DO 775 MB=1,5
  DO 771 I=1,20
    TEMP(I)=TMOM(MA,MB,I)
771 TMOM(MA,MB,I)=0.
  CALL STAS(TEMP,ISIG,20,KPUT,18)
  WRITE(6)(TEMP(I),ISIG(I),I=1,20)
  IF(18.LT.10)GO TO 775
  DO 772 I=1,20
    TMOM(MA,MB,I)=TEMP(I)
772 ITMOM(MA,MB,I)=ISIG(I)
775 CONTINUE
  DO 785 MA=1,5
  DO 785 MB=1,5
  DO 784 MB=1,5
    TEMP(MB)=BMOM(MA,MB)
784 BMOM(MA,MB)=0.
  CALL STAS(TEMP,ISIG,5,KPUT,18)
  WRITE(6)(TEMP(I),ISIG(I),I=1,5)
  IF(18.LT.10)GO TO 785
  DO 783 MB=1,5
    BMOM(MA,MB)=TEMP(MB)
783 ITMOM(MA,MB)=ISIG(MB)
785 CONTINUE
  IF(18.LT.10)GO TO 800
  PRINT 761
761 FORMAT(10/42X,'EMERGENT ENERGY DISTRIBUTION - TRANSMISSION*')
  I1=1
  I2=5
  I3=5
  IMAK=20
  IF(12.GT.IMAK)I2=IMAK
  PRINT 751,(THICK(I),I=1,12)
751 FORMAT(//1X,'THICKNESS(G/CM2)',5(E12.5,4X))
  PRINT 752
752 FORMAT(/8X,'E/E0')
  DO 753 K=1,20
753 PRINT 754,EFAC(K),(TPE(K,I),IFE(K,I),I=1,12)
754 FORMAT(8X,F4.2,2X,5(E12.5,14))
  IF(12.EQ.IMAK)GO TO 755
  I1=11*5
  I2=12*5
  GO TO 756
755 PRINT 757
757 FORMAT(//2X,'EMERGENT ENERGY DISTRIBUTION - BACKSCATTER*')
  PRINT 752
  DO 758 K=1,20
758 PRINT 759,EFAC(K),BFE(K),IBFE(K)
759 FORMAT(8X,F4.2,2X,E12.5,14)
  DO 760 I=1,IMAX
760 WRITE(2)IMAX,(EFAC(K),(TPE(K,I),IFE(K,I),IFE(K,I),K=1,20)
  WRITE(2)(EFAC(K),BFE(K),IBFE(K),K=1,20)
  PRINT 789
789 FORMAT(1X,'MOMENTS MATRICES - TRANSMISSION*,10X,'ENERGY (DOWN) -
  1 ANGLE(ACROSS)*')
791 FORMAT(//1X,'THICKNESS NO.',I3,5X,E12.5,' G/CM2*')
  DO 790 I=1,20
  IF(TTOT(I).EQ.0.0)GO TO 710
  DO 711 MA=1,5
  DO 711 MB=1,5
711 TMOM(MA,MB,I)=TMOM(MA,MB,I)/TTOT(I)
710 PRINT 791,I,THICK(I)
  DO 792 MB=1,5
792 PRINT 793,(TMOM(MA,MB,I),MA=1,5),(ITMOM(MA,MB,I),MA=1,5)

```

```

790 CONTINUE
793 F AT(1X,5E12.5,10X,515)
P .T 794
794 FORMAT(//1X, 'MOMENT MATRIX - - BACKSCATTER', 10X, 'ENERGY (DOWN) - -'
1ANGLE(ACROSS)')
DO 712 MA=1,5
DO 712 MB=1,5
712 BROW(MA,MB)=BROW(MA,MB)/BTOT(20)
DO 795 MB=1,5
795 PRINT 793, (BROW(MA,MB), MA=1,5), (BROW(MA,MB), MA=1,5)
800 CONTINUE
340 FORMAT(1X, 'LEGENDRE COEFFICIENTS - TRANSMISSION')
341 FORMAT(1X, 'LEGENDRE COEFFICIENTS - BACKSCATTER')
342 FORMAT(1X, 'ENERGY GROUP UPPER AND LOWER BOUNDS - ',
1F5.2, 'KEV TO', F5.2, 'KEV')
343 FORMAT(4X, 'THICKNESS', 12X, 'L', 7X, 'Q', 1X, '1', 12X, '2', 12X,
13X, '12X, 5')
344 FORMAT(1X, E12.5, 10X, F10.7, 3X, 5(F10.7, 13))
345 FORMAT(1X, 'TOTAL OF ALL ENERGIES')
IF(18.EQ.10) PRINT 499, NHIST
101 CONTINUE
220 FORMAT(///)
221 FORMAT(1X, F7.3, 1X, 10(F9.5, 13))
499 FORMAT(//1X, 'TOTAL NUMBER OF HISTORIES PROCESSED = ', I10)
Z=STO(1, LCHR)
DO 600 I=1, 20
IF(1Z.GT. THICK(1)) GO TO 600
IC=1
GO TO 601
600 CONTINUE
IF(1Z.LE.0.0) RETURN
IC=20
601 CHG(IC)=CHG(IC)+1./FNN
IF(1B.LT.10) RETURN
PRINT 602
DO 603 I=1, 20
603 PRINT 604, I, CHG(I)
602 FORMAT(//9X, 'CHARGE DEPOSITION PROFILE', 10X, 'ZONE', 10X, 'CHARGE')
604 FORMAT(11X, 12.9A, F9.7)
PRINT 714
714 FORMAT(///)
RETURN
END
SUBROUTINE LEP(Y,X,N)
C
C COMPUTE LEGENDRE POLYNOMIALS
C
DIMENSION Y(1)
Y(1)=1.
IF(N) 1,1,2
1 RETURN
2 Y(2)=X
IF(N-1) 1,1,3
3 DO 4 I=2,N
G=X*Y(I)
4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
RETURN
END
SUBROUTINE STATS(XVAL, ISIG, N, K, I8)
C
C COMPUTATION OF STATISTICAL STANDARD ERRORS (SEE VOL. I., P238.)
C
DIMENSION XVAL(1), ISIG(1), AVE(20), SIG(20)
DIMENSION ECAGE(4500), ECSIG(4500)
IF(18.NE.1) GO TO 2
DO 1 I=1,N

```

```

AVE(I) = XVAL(I)
1 5 I=1,99
1 5 I=0,0
10 DO 100 I=1,N
    INK=I-1
    ECALC(INK)=AVE(I)
    100 ECSVG(INK)=SIG(I)
    N=N+1
    RETURN
2 2 NSORS=IB
    DO 200 I=1,N
        INK=I-1
        AVE(I)=ECALC(INK)
    200 SIG(I)=ECSVG(INK)
    DO 3 I=1,N
        SIG(I)=(NSORS-2.0)/(NSORS-1.0)*SIG(I)+(XVAL(I)-AVE(I))*2/NSORS
        AVE(I)=(NSORS-1.0)*AVE(I)+XVAL(I)/NSORS
        XVAL(I) = AVE(I)
        ERROR=SQRT(SIG(I)/NSORS)
        ISIG(I)=99
        IF(AVE(I).EQ.0.0) GO TO 3
        ISIG(I)=100.0*ERROR/ABS(AVE(I))+.50001
        IF(ISIG(I).GT.99) ISIG(I)=99
    3 CONTINUE
    GO TO 10
END
SUBROUTINE ENERGY
COMMON/MC/NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,SPHIO,DELTE,
1EQ,N,IFLAG,MAXCOLL,LCHR,TLM,XO,YO,ZO,X,Y,Z,ECUT,THICK(21),DELZ,
2NMAX,ZMIN,ZMAX,INRAN,XIRA,NPRINT,TLAST,NBATCH,ENTAB(100),INCODE(3),
3XIRC,MAT,DNSTY(3),STO(4,101),TBO,DRB,RTOTAL,BCKF(3),NRA,NKRA,FORCE
4,THETA
DATA P12/1.570796327/
PM=PO-1
IF(FORCE.LT.0.)PM=-1.
IF(THETA.GT.P12)PM=-1.
EOLD=EGY
DO 1 I=1,100
1 IF(EOLD.GE.ENTAB(I))GO TO 2
I=100
2 IM=I
IF(I.EQ.1)IM=2
IM1=IM-1
STP=(ENTAB(IM1)-ENTAB(IM))/DRB
DELTE=DELTE-STP*PM*DELZ*CTH+FORCE*CTH*(DRB*PM*PQ*DELZ*CTH)
EGY=ENTAB(NCOLL)+DELTE
RETURN
END

```

Program Listing

- MCEL -

(Reference: Vol. I, Section IV.B.2)


```

PROGRAM MCCEL(INPUT,OUTPUT,TAPE2,TAPE3)

C C C PROGRAM MCCEL IS A SINGLE SCATTERING ANALOG MONTE CARLO PROGRAM TO
C C C SOLVE THE PROBLEM OF ELECTRON TRANSPORT RESULTING FROM AN ELECTRON
C C C BEAM SOURCE (20 - 50 KEV) INCIDENT ON A METAL SLAB. SCREENED
C C C RUTHERFORD SCATTERING IS ASSUMED. A USER OPTION IS ALSO INCLUDED
C C C TO ACCOUNT FOR THE EFFECT OF INELASTIC SCATTERING. (SEE VOL. 1,
C C C PAGES 231-239 FOR A DETAILED DESCRIPTION).
C C C COMMON/MC/STOI(7,601),NCOLL,CTWO,CTH,STH,PHI,CPHI,SPhi,EGY,CPHIIO,IC
C C C 1,SPHIQ,EO,SPAC,MBOA,N,IPLAG,WXCOLL,LCHR,STAB(200),ETAB(200),TLIM
C C C 2,KO,YO,ZO,X,Y,Z,EUCOT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
C C C 3,NPRINT,DNSTY,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XINC
C C C 4,ELND,WAT,DNSTY(2),ESAVE,INEL
C C C DIMENSION TIMR(11)

C C SUBROUTINE "SETURN" READS IN AND INITIALIZES THE VARIOUS PARAMETERS
C C NECESSARY FOR THE MONTE CARLO CALCULATION
C C CALL SETURN
C C PN=FLOAT(NMAX)
C C COLLNO=0.
C C MTM=1
C C CALL SECOND(TIME)
C C TIMR(1)=TIME
C C N=N+1

C C "SETHIS" INITIALIZES THE ELECTRON TRAJECTORY PARAMETERS AND IS
C C CALLED AT THE START OF EACH TRAJECTORY.
C C CALL SETHIS
C C "SCORE" RECORDS THE COORDINATES, DIRECTION AND ENERGY AT EACH
C C ELECTRON COLLISION SITE
C C CALL SCORE
C C "PENET" TRANSLATES THE ELECTRON FROM ONE COLLISION SITE TO THE
C C NEXT
C C 20 CALL PENET
C C IF(IFLAG.NE.0) GO TO 6
C C CALL ENERGY
C C IF(IFLAG.NE.0) GO TO 6
C C CALL SCORE

C C "ANGLES" COMPUTES THE POLAR AND AZIMUTHAL ANGLES OF THE ELECTRON
C C TRAJECTORY AFTER A COLLISION HAS OCCURRED.
C C CALL ANGLES
C C IF(IFLAG.NE.0)GO TO 6
C C GO TO 20
C C 6 CALL SCORE
C C CALL PROC
C C IF(NPRT.EQ.2)PRINT 2,(((STOK(L),K=1,7),L=1,LCHR)
C C COLLNO=COLLN0+FLOAT(NCOLL)/FN

C C THE CALCULATION IS ORGANIZED SO THAT THE TOTAL NUMBER(NMAX) OF
C C HISTORIES IS DIVIDED INTO 10 EQUAL PARTS. THIS SECTION OF CODE
C C COMPUTES THE AVERAGE TIME REQUIRED TO PROCESS EACH GROUP OF
C C (NMAX/10) HISTORIES AND DETERMINES WHETHER SUFFICIENT TIME HAS
C C BEEN ALLOWED TO FINISH THE CALCULATION. IF THERE IS NOT ENOUGH
C C TIME REMAINING, THE CALCULATION IS TERMINATED NORMALLY WITH THE
C C INTERMEDIATE RESULTS WRITTEN ON A PERMANENT FILE(TAPE3) BY

```

```

C
C
C
SUBROUTINE "PROC". IN THIS WAY THE CALCULATION CAN BE CONTINUED
* WITHOUT LOSING THE DATA OBTAINED THUS FAR DUE TO A COMPUTE TIME
OVER RUN.

IF(NBATCH.EQ.0)GO TO 60
NTEST=N-NBATCH*(N/NBATCH)
IF(NTEST.NE.0)GO TO 60
MTM=MTM+1
CALL SECOND(TIME)
TIMR(TIM)=TIME
DELT=TIME-TIMR(1)
AVGT=DELT/FLCAT(MTM-1)
TREM=TIM-TIM-DELT
TLEFT=1.5*AVGT
IPCT=10*(MTM-1)
PRINT 75,IPCT,TREM
75 FORMAT(1X,'THE PROBLEM IS',14,' PERCENT COMPLETE.',1X,
1*TIME TO FINISH IS',F12.3,'SECONDS')
IF(IPCT.EQ.100)GO TO 100
:IF(TREM.GE.TLEFT)GO TO 5
GO TO 100
60 IF(N-NMAX)5,100,100
2 FORMAT(1X,'E16.9)
100 PRINT 64,COLLG
64 CALL RANGET(IIRC)
IF(NBATCH.EQ.0)GO TO 200
MTM1=MTM-1
PRINT 65,MTM1,NBATCH
65 FORMAT(///1X,'THERE ARE ',14,' BATCHES OF',15,' HISTORIES//')
AVGT=0.
SUM=TIMR(1)
DO 66 M=2,MTM
TIMR(M)=TIMR(M)-SUM
SUM=SUM+TIMR(M)
AVGT=AVGT+TIMR(M)
MM=M-1
66 PRINT 67,MM,TIMR(M)
67 FORMAT(1X,'BATCH NO.',14,' TIME =',F8.3,' SECONDS')
AVGT=AVGT/FLCAT(MTM1)
PRINT 68,AVGT
68 FORMAT(///1X,'AVERAGE TIME PER BATCH =',F8.3,' SECONDS')
200 PRINT 190
190 FORMAT(6X,'INRAN',11X,'IRA',12X,'IRC')
PRINT 195,INRAN,XIRA,XIRC
195 FORMAT(3(1X,D16))
STOP
END
SUBROUTINE SETRUN
READ IN AND INITIALIZE ALL PARAMETERS NECESSARY FOR THE
CALCULATION.
COMMON/MC/STD(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,IC
1,SPHIO,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,1E,INRAN,XIRA
3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
DIMENSION INCODE(3),IK(11,2)
DATA CPHIO,SPHIO,PI,ZMIN,ZMAX,N/1,0,0,0,3,14159265,0,0,1000.,0/
DATA ECUT,SPAC,XO,YO,ZO/0.,1.,3,0./
DATA IK/1.,1.,5.,1.,5.,2.,4.,5.,7.,10.,15.,1000.,
1,3.,5.,7.,1.,5.,2.,2,5,3.,4.,5.,1000./
DATA DNSTY/2,7,19,3/
READ 1,NMAX,MAXCOLL,NPRNT

```



```

RETURN
C
C SUBROUTINE SETHIS
C
C INITIALIZE ELECTRON HISTORY
COMMON/MC/STO(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,IC
1,SPHIO,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELON,MAT,DNSTY(2),ESAVE,INEL
LCHR=0
NCOLL=0
PATH=0.
S=0.
X=XO
Y=YO
Z=ZO
CPHI=CPHIO
SPHI=SPHIO
PHI=0.0
CTH=CTHO=1.
STH=STO(1,-CTH**2)
EGY=EO
IE=1
CALL CROSS
IFLAG=0
RETURN
END
SUBROUTINE PENET
C
C CALCULATE INTER-COLLISION DISTANCE BY SAMPLING EXPONENTIAL
C ATTENUATION LAW AND CALCULATE COORDINATES OF COLLISION POINT
C
COMMON/MC/STO(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,IC
1,SPHIO,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELON,MAT,DNSTY(2),ESAVE,INEL
X1=X
Y1=Y
Z1=Z
4 RB=RANF(0)
IF(RB.GT.0.)GO TO 5
GO TO 4
5 S=-ALOG(RB)*AMBDA
Z=Z1+S*CTH
IF(Z.GT.ZMIN)GO TO 10
Z=ZMIN
S=(ZMIN-Z1)/CTH
IFLAG=1
10 IF(Z.LT.ZMAX)GO TO 11
Z=ZMAX
S=(ZMAX-Z1)/CTH
IFLAG=1
11 X=X1+S*STH*CPHI
Y=Y1+S*STH*SPHI
PATH=PATH+S
NCOLL=NCOLL+1
IF(NCOLL.EQ.MAXCOLL)IFLAG=1
RETURN
END
SUBROUTINE ESET
C
C SET UP A TABLE OF 100 SCREENED-RUTHERFORD CROSS SECTIONS AND
C STOPPING POWERS VS. ENERGY

```

```

C      ON/MC/STO(7.601),NCOLL,CTHO,CTH,STM,PHI,CPHI,SPHI,EGY( 410,IC
1,S=HIO,EO,SFAC,AMBOA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
    DIMENSION ZM(2),EXM(2),AM(2)
    DATA PI,XL2/3.14159265,.693147181/
    DATA AVCG,RO,FCC/6.025E23,2.8179E-13,
1,15360828/
    DATA ZM,AM,EXM/13..79..27..197..163.E-6,797.E-6/
    ZA=ZM(MAT)
    AA=AM(MAT)
    EXCIT=EXM(MAT)
    ENTAB(1)=EO
    DE=.01*EO
    DO 1 II=2,100
1 ENTAB(II)=ENTAB(II-1)-DE
    DO 3 II=1,100
3 ENTAB(II)=.511
    E1=(1.+E)*(1.+E)
    E2=E*(2.+E)
    BETA=SQRT(E2/E1)
    XLG=ALOG(E/E2)/(2.-(EXCIT/.511)**2))
    FM=1.-BETA*BETA*(.125*E-E-(2.+E+1.)*XL2)/E1
    STAB(11)=(FCC-ZA/AA/BETA**2)*(XLG+FM)
    ETAB(11)=0.5*(ZA*(1./3.)/121.245)**2/E2*(1.13+3.76*(ZA/137.))**2*
1E1/E2
    SIGTAB(11)=2.*PI*(1.+ZA)*AVCG*RO*RO*ZA/AA*E1/(E2*E2)/(ETAB(11)
1*(1.+S*ETAB(11)))
3 CONTINUE
    PRINT 75
75 FORMAT(1X,'CROSS SECTION AND STOPPING POWER TABLES',//1X,'*ENERGY(M
1EV) STOPPING POWER
    ETA
    SIGMA*//)
    DO 76 II=1,100
76 PRINT 77,ENTAB(II),STAB(II),ETAB(II),SIGTAB(II)
77 FORMAT(1X,F10.6,3E16.9)
    DO 50 II=1,100
50 IF(STAB(II).GT.0.0)GO TO 50
    IM=II-1
    GO TO 51
50 CONTINUE
    IM=100
51 ECUT=ENTAB(IM)
    RETURN
    END
    SUBROUTINE ANGLES
C      COMPUTE POLAR AND AZIMUTHAL ANGLES OF ELECTRON TRAJECTORY AFTER
C      COLLISION
C
    COMMON/MC/STO(7.601),NCOLL,CTHO,CTH,STM,PHI,CPHI,SPHI,EGY,CPHIO,IC
1,SPHIO,EO,SFAC,AMBOA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
    DATA TWOPI/6.283185307/
    RD=RNAN(0)
    IF(IC.EQ.1)GO TO 5
    CAY=EGY*ELOW/(EGY-ELOW)
    W=ELOW*CAY/(CAY-RD*ELOW)
    SOW=SQRT(W/EGY)
    COW=SQRT(1.-W/EGY)
    EGY=EGY-W*ESAVE
    IF(EGY.LT.ECUT)IFLAG=1
    GO TO 6

```

```

5 CPM=(2.*RD*(1.-ETA)-ETA)/(2.*RD+ETA)
6 SORT(1.-COM**2)
7 R=-RANF(0)
RHO=TWOPI*RC
CRHO=COS(RHO)
SRHO=SIN(RHO)
STH1=STH
CTH1=CTH
CTH=CTH1*COM+STH1*SOM+CRHO
STH=STH1*COM+STH1*SOM+CRHO
IF(STH1.EQ.0.0.OR.STH.EQ.0.0) GO TO 1
C1=(COM-CTH*CTH1)/(STH*STH1)
S1=SRHO*SOM/STH
CPH11=CPHI
SPH11=SPHI
CPH1=CPH11*C1-SPH11*S1
SPH1=CPH11*S1+SPH11*C1
GO TO 2
1 CPH1=CRHO
SPH1=SRHO
2 CONTINUE
PHI=ATAN2(SPH1,CPH1)
IF(PHI.LT.0.0)PHI=PHI+TWOPI
RETURN
END
SUBROUTINE SCORE
RECORD ELECTRON TRAJECTORY CHARACTERISTICS AND STORE IN BUFFER
ARRAY "STO"
COMMON/NC/STO(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPH10,IC
1,SPH10,ED,SPAC,AMBDA,N,IFLAG,MACCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
LCHR=LCHR+1
STO(1,LCHR)=Z
STO(2,LCHR)=CTHO
STO(3,LCHR)=CTH
STO(4,LCHR)=STH
STO(5,LCHR)=PHI
STO(6,LCHR)=EGY
STO(7,LCHR)=FLOAT(NCOLL)
IF(IFLAG.NE.0)STO(7,LCHR)=FLOAT(NCOLL)
RETURN
END
SUBROUTINE PROC
THIS SUBROUTINE PROCESSES THE ELECTRON TRAJECTORY DATA AND
COMPUTES THE TRANSMISSION AND BACKSCATTER FRACTION AND EMERGENT
FLUX VALUES FOR 10 SLAB THICKNESSES
THE HISTORIES ARE PROCESSED IN BATCHES OF 601 AT A TIME. THE ARRAY
"STO" IS USED AS THE BUFFER. IT IS FILLED IN SUBROUTINE "SCORE".
COMMON/NC/STO(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPH10,IC
1,SPH10,ED,SPAC,AMBDA,N,IFLAG,MACCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
DIMENSION ENERGY(20),ANGLE(5),TC(10,20),BC(10,20),TNM(11),
2TF(10,20),BF(10,20),TEMP(10),ISIG(10),SUM(10),ENERK(20)
DATA TC,TF,BC,BF/800*0./
DATA ANGLE/.2,.4,.6,.8,1./
IF(ITIME.GT.0)GO TO 1
ITIME=1

```

C C C

C C C C C C C

```

DO 24 I=1,11
24 T(I)=1.E+7*THICK(I)/DNSTY(MAT)
18...M=NHIST=0
REWIND 3
RTDEG=180./3.14159265
NNN=MMAX/10
DE=EO/20.
ENERGY(I)=EO-DE
ENERG(I)=1000.*ENERGY(I)
DO 2 K=2,20
ENERGY(K)=ENERGY(K-1)-DE
2 ENERK(K)=1000.*ENERGY(K)
1 WRITE(3)LCNR,1(STO(L1,L2),L1=1,7),L2=1,LCNR)
DO 101 L=1,LCNR
IF(STO(2,L)).EQ.0.0.AND.STO(6,L).EQ.0.0)STOP 5
IF(STO(7,L).NE.0.0)GO TO 5
NH=NH+1
NHIST=NHIST+1
TOLD=1
IZTOP=1
GO TO 101
5 Z=STO(1,L)
C
C
C
C CLASSIFY EMERGENT ENERGY
DO 25 K=1,20
25 IF(STO(6,L).GT.ENERG(K))GO TO 26
K=20
26 KC=K
C
C
C
C IF(Z.GT.ZMIN)GO TO 6
IF(IZTOP.GT.10)IZTOP=10
C
C BACKSCATTER FRACTION(BC) AND FLUX(BF)
DO 11 I=IZTOP,10
BC(I,KC)=BC(I,KC)+1.0
11 BF(I,KC)=BF(I,KC)+1./ABS(STO(3,L))
GO TO 102
6 DO 10 IZ=1,11
IF(Z.GT.THICK(IZ))GO TO 10
IF(IZ.EQ.1)GO TO 102
15=12
IF(15.LE.IZTOP)GO TO 102
IZTOP=15
10=15-1
C
C TRANSMISSION FRACTION(TC) AND FLUX(TF)
DO 12 IT=10LD,10
TC(IT,KC)=TC(IT,KC)+1.
12 TF(IT,KC)=TF(IT,KC)+1./STO(3,L)
10LD=IZTOP
GO TO 102
10 CONTINUE
102 IF(STO(7,L).LT.0.0)GO TO 112
GO TO 101
112 IF(NH.LT.NNN)GO TO 101
KPUT=1
NH=0
18=18+1
REWIND 2
FN=FLOAT(NNN)
DO 103 I=1,10
DO 103 K=1,20

```

```

TC(I,K)=TC(I,K)/PN
BC(I,K)=BC(I,K)/PN
TF(I,K)=TF(I,K)/PN
103 BF(I,K)=BF(I,K)/PN
    IF(18.EQ.10)CALL SECOND(TLAST)
    DO 150 NM=1,4
    DO 93 I=1,10
99 SUM(I)=0.
    IF(18.LT.10)GO TO 154
    PRINT 220
    GO TO (114,124,134,144),NM
114 PRINT 230
124 PRINT 231
134 PRINT 232
144 PRINT 233
153 PRINT 234,(THICK(I),I=1,10)
    PRINT 236,(TNM(I),I=1,10)
236 FORMAT(4X,NM=3X,10E12.5)
230 FORMAT(///45X,'TRANSMITTED CURRENT=')
231 FORMAT(///45X,'BACKSCATTERED CURRENT=')
232 FORMAT(///45X,'TRANSMITTED FLUX=')
233 FORMAT(///45X,'BACKSCATTERED FLUX=')
234 FORMAT(45X,'SLAB THICKNESS (G/CM2)=9X,10E12.5/3X,'E(KEV)=')
154 DO 104 K=1,20
    DO 105 I=1,10
    GO TO (115,125,135,145),NM
115 TEMP(I)=TC(I,K)
    TC(I,K)=0.
    GO TO 105
125 TEMP(I)=BC(I,K)
    BC(I,K)=0.
    GO TO 105
135 TEMP(I)=TF(I,K)
    TF(I,K)=0.
    GO TO 105
145 TEMP(I)=BF(I,K)
    BF(I,K)=0.
105 CONTINUE
    CALL STATS(TEMP,ISIG,10,KPUT,18)
    WRITE(2)ENERGY(K),(TEMP(I),ISIG(I),I=1,10)
    IF(18.LT.10)GO TO 104
    DO 156 I=1,10
156 SUM(I)=SUM(I)+TEMP(I)
    PRINT 221,ENERG(K),(TEMP(I),ISIG(I),I=1,10)
104 CONTINUE
    IF(18.EQ.10)PRINT 235,(SUM(I),I=1,10)
235 FORMAT(/2X,'TOTALS ',10(E12.5))
150 CONTINUE
    IF(18.EQ.10)PRINT 499,NHIST
101 CONTINUE
220 FORMAT(///)
221 FORMAT(1X,F7.3,1X,10(F9.5,13))
499 FORMAT(1X,'TOTAL NUMBER OF HISTORIES PROCESSED = ',I10)
    RETURN
    END
    SUBROUTINE STATS(XVAL,ISIG,N,K,18)
C
C "STATS" STORES THE OUTPUT QUANTITIES AND COMPUTES THE STANDARD
C STATISTICAL ERROR FOR EACH
C
    DIMENSION XVAL(1),ISIG(1),AVE(10),SIG(10)
    DIMENSION ECAGE(1000),ECSIG(1000)
    IF(18.NE.1) GO TO 2

```



```

DO 1 I=1,N
  X(I) = XVAL(I)
  SIG(I)=0.0
  DO 100 I=1,N
    IKK=K+I-1
    ECAGE(IKK)=AVE(I)
    EC SIG(IKK)=SIG(I)
  K=K+N
  RETURN
2 XNSORS=IB
DO 200 I=1,N
  IKK=K+I-1
  AVE(I)=ECAGE(IKK)
  SIG(I)=EC SIG(IKK)
DO 3 I=1,N
  SIG(I)=(XNSORS-2.0)/(XNSORS-1.0)*SIG(I)+(XVAL(I)-AVE(I))*2/XNSORS
  AVE(I) = ((XNSORS-1.0)*AVE(I)+XVAL(I))/XNSORS
  XVAL(I) = AVE(I)
  ERROR=SQRT(SIG(I)/XNSORS)
  ISIG(I)=99
  IF(AVE(I).EQ.0.0) GO TO 3
  ISIG(I)=100.0*ERROR/ABS(AVE(I))+.50001
  IF(ISIG(I).GT.99) ISIG(I)=99
3 CONTINUE
GO TO 10
END
SUBROUTINE ENERGY
C
C *ENERGY* PERFORMS A TABLE LOOK-UP ON THE ELECTRON ENERGY
C AND CHOOSES THE CORRESPONDING CROSS SECTION VALUE.
C
COMMON/MC/STO(7,601),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIQ,IC
1,SPHIQ,EO,SFAC,AMBDA,N,IFLAG,MACCOLL,LCHR,STAB(200),ETAB(200),TLIM
2,XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(1),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
4,ELOW,MAT,DNSTY(2),ESAVE,INEL
ETO=1000.*EO
EI=1000.*EGY
DD=100./ETO
IE=1+IFIX(DD*(ETO-ET))
DE=STAB(IE)*S
EGY=EGY-DE
IF(EGY.GT.ECUT)GO TO 4
IFLAG=1
EGY=ECUT
ENTRY CROSS
4 ETA=ETAB(IE)
IF(EGY.GT.ECUT)GO TO 5
IC=1
RETURN
5 IF(INEL.GT.0)GO TO 6
IC=1
AMBDA=1./SIGTAB(IE)
RETURN
C
C INELASTIC SCATTER
C
6 WAVG=EGY*ALOG(EGY/ELOW)
AMBIN=WAVG/STAB(IE)
SIGIN=1./AMBIN
SIGTOT=SIGIN+SIGTAB(IE)
AMBDA=1./SIGTOT
RA=RA+NF(0)
RATIO=SIGTAB(IE)*AMBDA
IC=2

```

C
C
C
C

ESAVE=DE
1 (A.LT.RATIO) IC=1
RL JRN
END

Program Listing

- BIGSLAB -

(Reference: Vol. I, Section IV.B.4)

```

PROGRAM BIGSLAB(INPUT,OUTPUT,TAPE2,TAPE3)

C PROGRAM "BIGSLAB" GENERATES TRANSMISSION AND ALBEDO DISTRIBUTIONS
C OF ELECTRONS SCATTERED OUT OF VERY THICK HOMOGENEOUS MEDIA(GOLD
C OR ALUMINUM). THE TRANSMISSION DATA GENERATED HERE ARE USED AS
C INPUT SOURCE FUNCTIONS FOR THE SECOND MONTE CARLO PROGRAM "INTFC"
C WHICH TREATS THE MATERIAL INTERFACE REGION. THE ALBEDO DATA
C GENERATED HERE ARE ALSO USED IN "INTFC" TO PROVIDE A BACKSCATTER
C DISTRIBUTION FROM THICK SLABS WHEN AN ELECTRON ESCAPES FROM THE
C INTERFACE REGION. IN THIS WAY A COSTLY MONTE CARLO CALCULATION IN
C THE THICK REGIONS, SURROUNDING THE MATERIAL INTERFACE REGION MAY
C BE AVOIDED. A LIST OF THE QUANTITIES CALCULATED BY "BIGSLAB" IS
C GIVEN IN VOL. 1, PAGES 253-257.

COMMON/MC/STO(7,51),NCOLL,CTHO,CTH,PMT,CPI,SPHI,EGY,CPHIO,
1SPHIO,ED,SPAC,AMBD,A,N,IFLAG,MACCOLL,LCHR,STAB(200),ETAB(200),TLIM,
20,VO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XINC
DIMENSION TIMR(11)

C SUBROUTINE "SETRUN" READS IN AND INITIALIZES THE VARIOUS PARAMETERS
C NECESSARY FOR THE MONTE CARLO CALCULATION

CALL SETRUN
FN=FLOAT(NMAX)
COLLNO=0.
MTM=1
CALL SECOND(TIME)
TIMR(1)=TIME
5 N=N+1

C "SETHIS" INITIALIZES THE ELECTRON TRAJECTORY PARAMETERS AND IS
C CALLED AT THE START OF EACH TRAJECTORY

CALL SETHIS

C "SCORE" RECORDS THE COORDINATES, DIRECTION AND ENERGY AT EACH
C ELECTRON COLLISION SITE.

CALL SCORE

C "PENET" TRANSLATES THE ELECTRON FROM ONE COLLISION SITE TO THE
C NEXT

20 CALL PENET

C THE VARIABLE "IFLAG >0" SIGNIFIES THE END OF THE CASE HISTORY.
C THIS MAY OCCUR VIA TRANSMISSION, BACKSCATTER, ENERGY BELOW CUT OFF
C OR COLLISION NUMBER EXCEEDING THE MAXIMUM ALLOWED NUMBER.

IF(IFLAG.NE.0) GO TO 6
CALL ENERGY
IF(IFLAG.NE.0) GO TO 6
CALL SCORE

C "ANGLES" COMPUTES THE POLAR AND AZIMUTHAL ANGLES OF THE ELECTRON
C TRAJECTORY AFTER A COLLISION HAS OCCURRED

CALL ANGLES
GO TO 20
6 CALL SCORE
CALL PROC
IF(NPRNT.EQ.2)PRINT 2,((STO(K,L),K=1,7),L=1,LCHR)

```

```

C C
C C      COLLNO=COLLNO+FLOAT(NCOLL)/FN
C C      YALCULATION IS ORGANIZED SO THAT THE TOTAL NUMBER(NMAX) F
C C      HISTORIES IS DIVIDED INTO 10 EQUAL PARTS. THIS SECTION OF CODE
C C      COMPUTES THE AVERAGE TIME REQUIRED TO PROCESS EACH GROUP OF
C C      (NMAX/10) HISTORIES AND DETERMINES WHETHER SUFFICIENT TIME HAS
C C      BEEN ALLOWED TO FINISH THE CALCULATION. IF THERE IS NOT ENOUGH
C C      TIME REMAINING, THE CALCULATION IS TERMINATED NORMALLY WITH THE
C C      INTERMEDIATE RESULTS WRITTEN ON A PERMANENT FILE(TAPE3) BY
C C      SUBROUTINE "PROC". IN THIS WAY THE CALCULATION CAN BE CONTINUED
C C      WITHOUT LOSING THE DATA OBTAINED THUS FAR DUE TO A COMPUTER TIME
C C      OVERRUN.
C C
C C      IF(NBATCH.EQ.0)GO TO 60
C C      NTEST=N-NBATCH*(N/NBATCH)
C C      IF(NTEST.LE.0)GO TO 60
C C      MTN=MTN+1
C C      CALL SECOND(TIME)
C C      IF(MTN.EQ.1)TIME=TLAST
C C      TIMR(MTN)=TIME
C C      DELT=TIME-TIMR(1)
C C      AVGT=DELT/FLOAT(MTN-1)
C C      TREMAIN=TLIM-DELT
C C      TLEFT=1.5*AVGT
C C      IPCT=10*(MTN-1)
C C      PRINT 75,IPCT,TREMAIN
C C      75 FORMAT(1X,'THE PROBLEM IS',I4,' PERCENT COMPLETE.',1X,
C C      'TIME TO FINISH IS',F12.3,'SECONDS')
C C      IF(IPCT.EQ.100)GO TO 100
C C      IF(TREMAIN.GE.TLEFT)GO TO 5
C C      GO TO 100
C C      60 IF(N-NMAX)5,100,100
C C      2 FORMAT(1X,'E16.9)
C C      100 PRINT 64,COLLNO
C C      CALL RANGE(XIRC)
C C      64 FORMAT(//1X,'AVERAGE NUMBER OF COLLISIONS PER HISTORY = ',F7.3)
C C      IF(NBATCH.EQ.0)GO TO 200
C C      MTN=MTN-1
C C      PRINT 65,MTN,NBATCH
C C      65 FORMAT(//1X,'THERE ARE ',I4,' BATCHES OF',I5,' HISTORIES')
C C      AVGT=0.
C C      SUM=TIMR(1)
C C      DO 66 M=2,MTN
C C      TIMR(M)=TIMR(M)-SUM
C C      SUM=SUM+TIMR(M)
C C      AVGT=AVGT+TIMR(M)
C C      MM=M-1
C C      66 PRINT 67,MM,TIMR(M)
C C      67 FORMAT(1X,'BATCH NO.',I4,' TIME = ',F8.3,' SECONDS')
C C      AVGT=AVGT/FLOAT(MTN)
C C      PRINT 68,AVGT
C C      68 FORMAT(//1X,'AVERAGE TIME PER BATCH = ',F8.3,' SECONDS')
C C      200 PRINT 190
C C      190 FORMAT(/6X,'INRAN=',11X,'IRA=',12X,'IRC=')
C C      PRINT 195,INRAN,XIRA,XIRC
C C      195 FORMAT(3(1X,O16))
C C      STOP
C C      END
C C      SUBROUTINE SETRUN
C C      READ IN AND INITIALIZE ALL PARAMETERS NECESSARY FOR THE
C C      CALCULATION
C C
C C      COMMON/NC/STD(7,51),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
C C      1SPHIO,EO,SPAC,AMBDA,N,IFLAG,NMAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
C C      2XD,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA

```

```

3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),YIRC
D1( SIGN INCODE(3)
DATA CPHIO,SPHIO,PI,ZMIN,ZMAX,N/1.0,0.0,0.3,14159265,0.0,1000.,,0./
DATA ECUT,SFAC,XO,YO,ZO/0.,800.,,3*0./
DATA THICK/2.5,5.,7.5,10.,12.5,15.,17.5,20.,25.,50.,10000./
READ 1,NMAX,MAXCOLL,NPRINT,INPT,NSLABS
C
C NMAX - TOTAL NO. OF MONTE CARLO HISTORIES
C STORED ON TAPE OR DISK UNTIL PROCESSING TIME
C MAXCOLL - MAXIMUM NUMBER OF COLLISIONS PER HISTORY TO BE ALLOWED
C NPRINT - DEBUG PRINT CONTROL PARAMETER
C NPRINT EQUAL TO ZERO SUPPRESSES DEBUG PRINTOUT
C NPRINT NOT EQUAL TO ZERO ACTIVATES DEBUG PRINTOUT
C INPT(0/1) - (DEFAULT/USER PROVIDED) VALUES FOR INITIAL RANDOM
C NUMBER "INRAN". STOPPING POWER FUDGE FACTOR "SFAC".
C AND INITIAL PARTICLE COORDINATES "XO,YO,ZO".
C
C
READ 3,TLIM,CTHO,EO,ECUT
IF(NSLABS.EQ.0)GO TO 50
READ 3,THICK(1),I=1,11)
50 DO 51 I=1,11
51 THICK(I)=THICK(I)*1.E-5
IF(INPT.GT.0)GO TO 40
INRAN=0
GO TO 45
40 READ 1,((INCODE(1),I=1,3)
IF(INCODE(1).EQ.1)READ 8,INRAN
IF(INCODE(2).EQ.1)READ 3,SFAC
IF(INCODE(3).EQ.1)READ 3,XO,YO,ZO
C
C EO - INITIAL PARTICLE ENERGY
C XO,YO,ZO - INITIAL PARTICLE COORDINATES
C SFAC - STOPPING POWER FUDGE FACTOR
C
45 PRINT 4,NMAX,MAXCOLL
PRINT 9,EO,ECUT,XO,YO,ZO,CTHO,CPHIO,SPHIO
PRINT 11,SPAC
11 FORMAT(1X,'STOPPING POWER FUDGE FACTOR (SFAC) =',F10.5)
9 FORMAT(1X,'EO (INITIAL ENERGY MEV)',E12.5//1X,'ECUT (LOW ENERGY
ICUT OFF MEV) ',E12.5//1X,'XO, YO, ZO, (SOURCE POINT COORDINATES)
3 ',E12.5//1X,'CTHO (INCIDENT POLAR COSINE) ',E12.5//1X,'CPHIO
4,SPHIO (INCIDENT AZIMUTH) ',E12.5)
6 FORMAT(018)
1 FORMAT(6110)
3 FORMAT(BF10.0)
4 FORMAT(1//1X,'NMAX (NO. OF HISTORIES) ',E16//1X,'MAXCOLL (MAXIMUM
1 ALLOWED NUMBER OF COLLISIONS) ',I3)
10 N=0
XIRA=XIRC=INRAN
NBATCH=NMAX/10
CALL RANSET(XIRA)
CALL ESET
RETURN
END
SUBROUTINE SETHIS
C
C INITIALIZE ELECTRON HISTORY
C
COMMON/MC/STO(17,51),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,EO,SFAC,AMBD,A,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC
LCHR=0
NCOLL=0
PATH=0.
S=0.

```



```

      BETASQRT(E2/E1)
      X(1)=LOG((E-E2)/(2.0*(EXCIT/.511)**2))
      FM...BETA=BETA*(1.125+E-12.0+E+1.0)*XL2/E1
      STAB(11)=(FCC*ZA/AA/BETA**2)*(XLG+FM)*SFAC
      ETAB(11)=0.5*(ZA**2/(1.3.)/121.245)**2/E2*(1.13+3.76*(ZA/137.0)**2
      1E1/E2)
      SIGTAB(11)=ZA*(1.4*ZA)*AVOG*RO*RO*ZA/AA*E1/(E2*E2)/(ETAB(11)
      1*11+.5*ETAB(11)))
      ETAB(11)=.035276
      3 IF(11.GT.91)ETAB(11)=.103378
      IF(NPRNT.EQ.0)RETURN
      PRINT 75
      75 FORMAT(1X,'CROSS SECTION AND STOPPING POWER TABLES',//,1X,'ENERGY(M
      1EV) STopping POWER ETA
      SIGMA//)
      DO 76 11=1,191
      76 PRINT 77,ETAB(11),STAB(11),ETAB(11),SIGTAB(11)
      77 FORMAT(1X,F10.6,3E16.9)
      RETURN
      END
      SUBROUTINE ANGLES
      C
      C COMPUTE POLAR AND AZIMUTHAL ANGLES OF ELECTRON TRAJECTORY AFTER
      C COLLISION
      C
      COMMON/NC/STO(7,51),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHI0,
      1SPHI0,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
      2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
      3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATN,S,SIGTAB(200),ENTAB(200),XIRC
      DATA TPOI/6.283185307/
      RD=2.*RANF(0)
      COM=(RD*(1.4*ETA)-ETA)/(RD+ETA)
      SCM=SQRT(1.-COM**2)
      RC=RANF(0)
      RHO=TPOI*RC
      CRMO=COS(RHO)
      SRMO=SIN(RHO)
      STH=CTH
      CTH=CTH
      CTH=CTH1*COM+STM1*SDM+CRMO
      STH=SQRT(1.-CTH**2)
      IF(STM1.EQ.0.0.OR.STH.EQ.0.0) GO TO 1
      C1=(COM-CTH*CTH1)/(STM*STM1)
      S1=SRMO*SCM/STH
      CPHI1=CPHI
      SPHI1=SPHI
      CPHI=CPHI1+C1-SPHI1*S1
      SPHI=CPHI1+S1+SPHI1*C1
      GO TO 2
      1 CPHI=CRMO
      SPHI=SRMO
      2 CONTINUE
      PHI=ATAN2(SPHI,CPHI)
      IF(PHI.LT.0.0)PHI=PHI+TPOI
      RETURN
      END
      SUBROUTINE SCORE
      C
      C RECORD ELECTRON TRAJECTORY CHARACTERISTICS IN BUFFER ARRAY "STO"
      C
      COMMON/NC/STO(7,51),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHI0,
      1SPHI0,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
      2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
      3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATN,S,SIGTAB(200),ENTAB(200),XIRC
      LCHR=LCHR+1
      STO(1,LCHR)=Z

```



```

300 STO(2, LCHR) = CTHO
301 SIF(3, LCHR) = CTH
302 SIF(4, LCHR) = STM
303 SIF(5, LCHR) = PHI
304 STO(6, LCHR) = EGY
305 STO(7, LCHR) = FLOAT(NCOLL)
306 IF(FLAG.NE.0) STO(7, LCHR) = -FLOAT(NCOLL)
307 RETURN
308 END
309 SUBROUTINE PROC
310
311 PROCESS TRAJECTORY DATA. FOR LIST OF COMPUTED QUANTITIES, SEE
312 VOL. I, PAGES 253-257. ALL COMPUTED QUANTITIES ARE PRINTED OUT
313 WITH THEIR STATISTICAL STANDARD ERRORS. THE TRANSMISSION AND AL-
314 BEDO SPHERICAL HARMONICS COEFFICIENTS ARE WRITTEN OUT TO A PERMA-
315 NENT FILE (TAPE2).
316
317 COMMON/MC/STO(7,51), NCOLL, CTHO, CTH, STM, PHI, CPHI, SPHI, EGY, CPHIO,
318 1SPHIO, EO, SFAC, AMBDA, N, IFLAG, MAXCOLL, LCHR, STAB(200), ETAB(200), TLM,
319 2XO, YO, ZO, X, Y, Z, ECUT, CLAB, THICK(11), NMAX, ZMIN, ZMAX, IE, INRAN, XIRA
320 3, NPRT, TLAST, CTHC, NBATCH, ETA, PATH, 5, SIGTAB(200), ENTAB(200), XIRC
321 DIMENSION TRANC(10,5,20), BACAC(10,5,20), TRANF(10,5,20), BACKF(10,5,
322 120), ENERGY(20), ANGLE(5), TC(10,20), BC(10,20), CMEAN(20), NKC(20),
323 2TF(10,20), BF(10,20), TEMP(10), ISIG(10), SUM(10), SCMEAN(20), ECRSE(10)
324 DIMENSION ALBEDO(10,10,12), CURTZ(20), CTFINE(10), PHFINE(12), SIGX(20),
325 1SIGZ(20), CURTX(20), CURTZ(20), EXPCOF(10,10), PL(10), ISG(10,10,12)
326 DIMENSION CMC(10), SMC(10), NKCR(10), SIGZC(10), SIGXC(10),
327 1CURTAC(10), CURTZC(10)
328 DIMENSION APLUS(9,9,10), AMINUS(9,9,10), YCOF(9,9), CP(9),
329 1SP(9), PLM(9,9), ISGAP(9,9,10), ISGAM(9,9,10), ISGEX(10,10)
330 DIMENSION TPLUS(9,9,10), TMINUS(9,9,10), ITP(9,9,10), ITM(9,9,10)
331 EQUIVALENCE(APLUS(1), TPLUS(1)), (AMINUS(1), TMINUS(1))
332 EQUIVALENCE(ISGAP(1), ITP(1)), (ISGAM(1), ITM(1))
333 EQUIVALENCE(SIGZC(1), BF(1,1)), (SIGXC(1), BF(1,2))
334 EQUIVALENCE(CURTZC(1), BF(1,3)), (CURTAC(1), BF(1,4))
335 EQUIVALENCE(ALBEDO(1), TRANC(1))
336 EQUIVALENCE(ALBEDO(100), TRANF(1))
337 EQUIVALENCE(ISG(1), BACAC(1))
338 EQUIVALENCE(SIG(100), BACKF(1))
339 EQUIVALENCE(SIGZ(1), TF(1,1)), (SIGX(1), TF(1,3))
340 EQUIVALENCE(CURTZ(1), TF(1,5)), (CURTX(1), TF(1,7))
341 EQUIVALENCE(CTFINE(1), TF(1,9)), (PHFINE(1), TF(1,10))
342 EQUIVALENCE(ECRSE(1), TF(1,12))
343 DATA CMEAN, SCMEAN, NKC/40.0, 20.0/
344 DATA CMC, SMC, NKCR/20.0, 10.0/
345 DATA ZMIN/0./
346 DATA ANGLE/2.4, 6.8, 1./
347 DATA YCOF, PLM/162.0/
348 DATA TRANC, BACAC, TRANF, BACKF/4000.0./
349 DATA TPLUS, TMINUS, EXPCOF, ITP, ITM/1720.0, 1620.0/
350 DATA ITIME/0/
351 IF(11*TIME.GT.0) GO TO 1
352 DO 30 LI=1,9
353 FL=(FLOAT(LI)*2.+1.)/6.283185307
354 DO 31 MI=1,LI
355 LM=L1+MI-1
356 LM=L1+MI
357 FI=1.
358 DO 32 I=LMI,LM
359 FI=FI+FLOAT(I)
360 YCOF(LI,MI)=SORT(FL/FI)
361 CONTINUE
362 ITIME=1
363 IB=NH+NHIST-0
364 REWIND 3
365 RTDEG=180./3.14159285

```

```

NNN=MAX/10
FN=LOAT(NNN)
DO 10 /20
  ENERGY(1)=ED-DE
  DO 2 K=2,20
    2 ENERGY(K)=ENERGY(K-1)-DE
    1 WRITE(3)LCNR,((STO(L1,L2),L1=1,7),L2=1,LCNR)
    DO 101 L=1,LCNR
      IF(STO(2,L).EQ.0.0.AND.STO(6,L).EQ.0.0)STOP 5
      IF(STO(7,L).NE.0.0)GO TO 5
      NH=NH+1
      NHIST=NHIST+1
      IOLD=1
      IZTOP=1
      GO TO 101
    5 Z=STO(1,L)
  C
  C CLASSIFY EMERGENT ANGLE
  C
  DO 15 J=1,5
    15 IF(ABS(STO(3,L)).LT.ANGLE(J))GO TO 16
    J=5
    16 JC=J
  C
  C CLASSIFY EMERGENT ENERGY
  C
  DO 25 K=1,20
    25 IF(STO(6,L).GT.ENERGY(K))GO TO 26
    K=20
    26 KC=K
  C
  C
  IF(IZ.GT.ZMIN)GO TO 6
  IF(IZTOP.GT.10)IZTOP=10
  DO 11 I=IZTOP,10
    CTC=ABS(STO(3,L))
    BACKC(1,JC,KC)=BACKC(1,JC,KC)+1.0
    11 BACKF(1,JC,KC)=BACKF(1,JC,KC)+1./ABS(STO(3,L))
    CPH=COS(STO(5,L))
    CMEAN(KC)=CMEAN(KC)+CTC
    SCMEAN(KC)=SCMEAN(KC)+CPH*STO(4,L)
    NKC(KC)=NKC(KC)+1
    KCR=(KC+1)/2
    CMC(KCR)=CMC(KCR)+CTC
    SMC(KCR)=SMC(KCR)+CPH*STO(4,L)
    NKCR(KCR)=NKCR(KCR)+1
    GO TO 102
  6 DO 10 IZ=1,11
    IF(IZ.GT.THICK(IZ))GO TO 10
    IF(IZ.EQ.1)GO TO 102
    IS=IZ
    IF(15.-LE.IZTOP)GO TO 102
    IZTOP=IS
    IO=IS-1
    DO 12 IT=IOLD,IO
      TRANC(IT,JC,KC)=TRANC(IT,JC,KC)+1.
      12 TRANF(IT,JC,KC)=TRANF(IT,JC,KC)+1./STO(3,L)
      IOLO=IZTOP
      IF(10.-NE.10)GO TO 102
      CTHET=STO(3,L)
      STHET=STO(4,L)
      PHI=STO(5,L)
      CALL LEP(PL,CTHET,9)
      KCR=(KC+1)/2
      DO 1430 LL=1,10
        1430 EXPCOF(LL,KCR)=EXPCOF(LL,KCR)+PL(LL)/FN

```

```

DO 1338 M=1,9
  CP( )=COS(M*PHI)
1338 SP(M)=SIN(M*PHI)
  CALL ASSOC(PL,CTHET,STNET,PLM)
  DO 1341 LL=1,9
    DO 1339 M=1,LL
      FM=YCOF(LL,M)*PLM(LL,M)/FN
      TPLUS(LL,M,KCR)=TPLUS(LL,M,KCR)+FM*CP(M)
      TMINUS(LL,M,KCR)=TMINUS(LL,M,KCR)+FM*SP(M)
1341 CONTINUE
  GO TO 102
10 CONTINUE
102 IF(STO(7,L).LT.0.0)GO TO 112
  GO TO 101
112 IF(NH.LT.NNN)GO TO 101
  KPUT=1
  NH=0
  IB=IB+1
  REWIND 2
  DO 103 I=1,10
    DO 103 K=1,20
      TC(I,K)=BC(I,K)=TF(I,K)=BF(I,K)=0.
    DO 103 J=1,5
      TC(I,K)=TC(I,K)+TRANC(I,J,K)/FN
      BC(I,K)=BC(I,K)+BACKC(I,J,K)/FN
      TF(I,K)=TF(I,K)+TRANF(I,J,K)/FN
      BF(I,K)=BF(I,K)+BACKF(I,J,K)/FN
103 TRANC(I,J,K)=BACKC(I,J,K)=TRANF(I,J,K)=BACKF(I,J,K)=0.0
  IF(IB.EQ.10)CALL SECOND(LAST)
  DO 150 MM=1,4
    DO 93 I=1,10
      SUM(I)=0.
      IF(IB.LT.10)GO TO 154
      PRINT 220
      GO TO (114,124,134,144),MM
114 PRINT 230
124 PRINT 231
134 PRINT 232
144 PRINT 233
153 PRINT 234,(THICK(I),I=1,10)
230 FORMAT(//45X,'TRANSMITTED CURRENT=/')
231 FORMAT(//45X,'BACKSCATTERED CURRENT=/')
232 FORMAT(//45X,'TRANSMITTED FLUX=/')
233 FORMAT(//45X,'BACKSCATTERED FLUX=/')
154 DO 104 K=1,20
    DO 105 I=1,10
      GO TO (115,125,135,145),MM
115 TEMP(I)=TC(I,K)
  GO TO 105
125 TEMP(I)=BC(I,K)
  GO TO 105
135 TEMP(I)=TF(I,K)
  GO TO 105
145 TEMP(I)=BF(I,K)
105 CONTINUE
  CALL STATS(TEMP,ISIG,10,KPUT,IB)
  WRITE(2)ENERGY(K),(TEMP(I),ISIG(I),I=1,10)
  IF(IB.LT.10)GO TO 104
  DO 156 I=1,10
156 SUM(I)=SUM(I)+TEMP(I)
  PRINT 221,ENERGY(K),(TEMP(I),ISIG(I),I=1,10)
104 CONTINUE

```

```

IF(18.EQ.10)PRINT 235,(SUM(I),I=1,10)
235 FOR I=1,2X,*TOTALS *,10(E12.5)
150 CD NUC
DO 1700 I=1,9
DO 1700 J=1,9
DO 1701 K=1,10
TEMP(K)=TPUS(I,J,K)
1701 TPUS(I,J,K)=0.
CALL STATS(TEMP,ISIG,10,KPUT,18)
IF(18.LT.10)GO TO 1700
DO 1702 K=1,10
TPUS(I,J,K)=TEMP(K)
1702 ITP(I,J,K)=ISIG(K)
1700 CONTINUE
DO 1710 I=1,9
DO 1710 J=1,9
DO 1711 K=1,10
TEMP(K)=TMINUS(I,J,K)
1711 TMINUS(I,J,K)=0.
CALL STATS(TEMP,ISIG,10,KPUT,18)
IF(18.LT.10)GO TO 1710
DO 1712 K=1,10
TMINUS(I,J,K)=TEMP(K)
1712 ITM(I,J,K)=ISIG(K)
1710 CONTINUE
DO 1720 I=1,10
AL=1
FAL=SQRT((2.*AL-1.)/12.56637061)
DO 1721 J=1,10
TEMP(J)=EXPCOF(I,J)
1721 EXPCOF(I,J)=0.
CALL STATS(TEMP,ISIG,10,KPUT,18)
IF(18.LT.10)GO TO 1720
DO 1722 J=1,10
EXPCOF(I,J)=FAL*TEMP(J)
1722 ISGEX(I,J)=ISIG(J)
1720 CONTINUE
IF(18.EQ.10)PRINT 499,NHIST
101 CONTINUE
IF(NHIST.LT.NMAX)RETURN
220 FORMAT(///)
221 FORMAT(1X,F7.3,1X,10(F9.5,13))
499 FORMAT(1X,*TOTAL NUMBER OF HISTORIES PROCESSED = *,110)
PRINT 179
DO 180 K=1,20
IF(NKC(K).EQ.0)GO TO 180
CMEAN(K)=CMEAN(K)/FLOAT(NKC(K))
SCMEAN(K)=SCMEAN(K)/FLOAT(NKC(K))
180 PRINT 181,ENERGY(K),CMEAN(K),SCMEAN(K)
179 FORMAT(///10X,*AVERAGE COS(THETA)=,5X,*AVERAGE SIN(THETA)COS(PHI)*
1/3X,*E(MEV)=)
181 FORMAT(1X,F7.3,2X,E12.5,14X,E12.5)
PRINT 179
DO 185 K=1,10
FDE=FLOAT(2*K)*DE
ECRSE(K)=EO-FDE
IF(NKCR(K).EQ.0)GO TO 185
CMC(K)=CMC(K)/FLOAT(NKCR(K))
SMC(K)=SMC(K)/FLOAT(NKCR(K))
185 PRINT 181,ECRSE(K),CMC(K),SMC(K)
PRINT 1823
1823 FORMAT(///45X,*LEGENDRE COEFFICIENTS OF TRANSMISSION*/
14X,*E(MEV) L=,4X,*0=,11X,*1=,11X,*2=,11X,*3=,11X,*4=,
211X,*5=,11X,*6=,11X,*7=,11X,*8=,11X,*9=)
DO 1826 K=1,10
1826 PRINT 526,ECRSE(K),(EXPCOF(LL,K),LL=1,10)

```

```

      651
      DO 525 K=1,10
1525 PRINT 524,ECRSE(K),((ISGEX(LL,K),LL=1,10)
      DO 1345 KE=1,10
      PRINT 1347,ECRSE(KE)
1347 FORMAT(/IX,ENERGY = ,F10.5, ,MEV*/IX,SPHERICAL HARMONICS COEFF
      ICIENTS FOR TRANSMISSION, TPLUS*/)
      DO 1343 L=1,9
1343 PRINT 349,L,((TPLUS(L,M,KE),M=1,9)
      PRINT 651
      DO 1652 L=1,9
1652 PRINT 653,L,((ITP(L,M,KE),M=1,9)
      PRINT 1348
1348 FORMAT(IX,SPHERICAL HARMONICS COEFFICIENTS FOR TRANSMISSION, TMIN
      IUS*/)
      DO 1342 L=1,9
1342 PRINT 349,L,((TMINUS(L,M,KE),M=1,9)
      PRINT 651
      DO 1654 L=1,9
1654 PRINT 653,L,((ITM(L,M,KE),M=1,9)
      WRITE(2)(EXPCOF(L,KE),L=1,10)
      DO 1251 L=1,9
      WRITE(2)(TPLUS(L,M,KE),M=1,9)
      WRITE(2)(TMINUS(L,M,KE),M=1,9)
1351 CONTINUE
1345 CONTINUE
      REWIND 3
      IB=NH=0
      DO 300 I=1,10
      DO 337 L=1,9
      DO 327 M=1,9
337 APLUS(L,M,I)=AMINUS(L1,M1,I)=0.
      SIGZ(I)=SIGAC(I)=CURTZX(I)=CURTXC(I)=0.
      CTFINE(I)=0.1*FLOAT(I)
      DO 300 J=1,10
      DO 300 K=1,12
300 ALBEDO(I,J,K)=0.
      DO 289 K=1,12
      DO 289 PHFINE(K)=30.*FLOAT(K)
      DO 302 K=1,20
302 SIGX(K)=SIGZ(K)=CURTX(K)=CURTZ(K)=0.
      DO 312 K=1,10
      DO 312 L=1,10
      ISGEX(I,K)=0.
312 EXPCOF(I,K)=0.
298 READ(3)LCRR,((STO(L1,L),L1=1,7),L=1,LCRR)
      IF(EOF(3))500,299
299 NH=NH+1
      DO 301 L=2,LCRR
      IF(STO(L1,L).GT.ZMIN)GO TO 301
      CTHET=ABS(STO(3,L))
      STHET=STO(4,L)
      PHI=STO(5,L)
      CLASSIFY EMERGENT POLAR ANGLE
      DO 315 J=1,10
315 IF(CTHET.LT.CTFINE(J))GO TO 316
      J=10
316 JC=J
      CLASSIFY EMERGENT ENERGY
      DO 325 I=1,10
325 IF(STO(6,L).GT.ECRSE(I))GO TO 326
      I=10
326 IC=I
      CLASSIFY AZIMUTH
      PP=RTDEG*PHI
      NO 225 M=1,12

```

```

335 IF (NH.LT.NNN)GO TO 336
336 K=1
C ALBEDO(IC,JC,KC)=ALBEDO(IC,JC,KC)+1./FN
C COMPUTE VARIANCE,STD. DEV. AND SKEWNESS OF DIRECTION COSINES
C ON FINE ENERGY GRID
C CLASSIFY ENERGY ON FINE GRID
DO 425 K=1,20
425 IF (STO(6,L).GT.ENERGY(K))GO TO 426
K=20
426 KE=K
DIF=CTHET-CMEAN(KE)
SIGZ(KE)=SIGZ(KE)+DIF*DIF
CURTZ(KE)=CURTZ(KE)+DIF*DIF*DIF
DIF=STHET-COS(PHI)-SCMEAN(KE)
SIGX(KE)=SIGX(KE)+DIF*DIF
CURTX(KE)=CURTX(KE)+DIF*DIF*DIF
CLASSIFY ENERGY ON COARSE GRID
KCRSE=(KE+1)/2
DIF=CTHET-CMC(KCRSE)
SIGZC(KCRSE)=SIGZC(KCRSE)+DIF*DIF
CURTZC(KCRSE)=CURTZC(KCRSE)+DIF*DIF*DIF
DIF=STHET-COS(PHI)-SMC(KCRSE)
SIGXC(KCRSE)=SIGXC(KCRSE)+DIF*DIF
CURTXC(KCRSE)=CURTXC(KCRSE)+DIF*DIF*DIF
GET LEGENDRE COEFFICIENTS OF POLAR ANGULAR DISTRIBUTION
CALL LEPI(PL,CTHET,9)
DO 430 LL=1,10
430 EXPCOF(LL,KCRSE)=EXPCOF(LL,KCRSE)+PL(LL)/FN
CP(M)=COS(M*PHI)
SP(M)=SIN(M*PHI)
338 CALL ASSOC(PL,CTHET,STHET,PLM)
DO 341 LL=1,9
431 FM=YCOF(LL,M)*PLM(LL,M)
FM=FM/FN
APLUS(LL,M,KCRSE)=APLUS(LL,M,KCRSE)+FM*CP(M)
339 AMINUS(LL,M,KCRSE)=AMINUS(LL,M,KCRSE)+FM*SP(M)
341 CONTINUE
301 CONTINUE
IF (NH.LT.NNN)GO TO 298
NH=0
18=18+1
KPUT=1
DO 600 K=1,12
DO 600 J=1,10
DO 601 I=1,10
TEMP(I)=ALBEDO(I,J,K)
601 ALBEDO(I,J,K)=0.
CALL STATS(TEMP,ISIG,10,KPUT,18)
IF (18.LT.10)GO TO 600
DO 602 I=1,10
ALBEDO(I,J,K)=TEMP(I)
602 ISG(I,J,K)=ISIG(I)
600 CONTINUE
DO 700 I=1,9
DO 701 K=1,10
TEMP(K)=APLUS(I,J,K)
701 APLUS(I,J,K)=0.
CALL STATS(TEMP,ISIG,10,KPUT,18)
IF (18.LT.10)GO TO 700
DO 702 K=1,10
APLUS(I,J,K)=TEMP(K)
702 ISGAP(I,J,K)=ISIG(K)

```

```

700 CONTINUE
DO 0 I=1,9
DO 10 J=1,9
DO 711 K=1,10
TEMP(K)=AMINUS(I,J,K)
711 AMINUS(I,J,K)=0.
CALL STATST(TEMP,ISIG,10,KPUT,18)
IF(18.LT.10)GO TO 710
DO 712 K=1,10
AMINUS(I,J,K)=TEMP(K)
712 ISGAM(I,J,K)=ISIG(K)
710 CONTINUE
DO 720 I=1,10
DO 721 J=1,10
TEMP(J)=EXPCOF(I,J)
721 EXPCOF(I,J)=0.
CALL STATST(TEMP,ISIG,10,KPUT,18)
IF(18.LT.10)GO TO 720
DO 722 J=1,10
EXPCOF(I,J)=TEMP(J)
722 ISGEX(I,J)=ISIG(J)
720 CONTINUE
GO TO 298
C
NORMALIZE AND PRINT
500 DO 510 I=1,10
DO 501 J=1,10
SUM(J)=0.
DO 511 K=1,12
511 SUM(J)=SUM(J)+ALBEDO(I,J,K)
501 CONTINUE
ELDW=0.
EHIGH=EO
IF(1.GT.1)ENIGH=ECRSE(I-1)
PRINT 502,EO,ENIGH,ECRSE(I),(CTFINE(J),J=1,10)
502 FORMAT(//20X,'ALBEDOS --- INCIDENT ENERGY =',F7.3,'MEV',
15X,'EMERGENT ENERGY BOUNDS =',F7.3,' TO ',F7.3,' MEV',
245X,'EMERGENT POLAR ANGLE COSINE=9X,10E12.5/1X,PHI(DEG)=')
DO 503 K=1,12
503 PRINT 504,PHFINE(K),(ALBEDO(I,J,K),ISG(I,J,K),J=1,10)
504 FORMAT(1X,F7.3,1X,10(F9.5,13))
510 CONTINUE
C
FINAL COMPUTATION OF VARIANCES, SKEWNESS AND LEGENDRE COEFFS
PRINT 519
DO 520 KE=1,20
STDX=STDZ=GAMX=GAMZ=0.
IF(NKCR(KE).EQ.0)GO TO 520
FNK=FLOAT(NKCR(KE))
SIGZ(KE)=SIGZ(KE)/FNK
CURTZ(KE)=CURTZ(KE)/FNK
SIGX(KE)=SIGX(KE)/FNK
CURTX(KE)=CURTX(KE)/FNK
STDZ=SQRT(SIGZ(KE))
STDX=SQRT(SIGX(KE))
GAMX=CURTZ(KE)/(STDZ**3)
GAMZ=CURTZ(KE)/(STDZ**3)
1STDZ,CURTZ(KE),GAMX
520 PRINT 521,ENERGY(KE),SIGZ(KE),STDZ,CURTZ(KE),GAMZ,SIGX(KE),
19X,'STDZ',8X,'KURTOS',8X,'SKWZ',8X,'VARX',8X,'STDZ',8X,'KURTOS',
26X,'SKWZ')
521 FORMAT(1X,F7.3,1X,10E12.5)
DO 620 K=1,10
STDX=STDZ=GAMX=GAMZ=0.
IF(NKCR(K).EQ.0)GO TO 620

```



```

3 DOX I=2,N
4 G=X(I)
5 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
RETURN
END
SUBROUTINE STATS(XVAL,ISIG,N,M,IB)
C
C COMPUTATION OF STATISTICAL STANDARD ERRORS (SEE VOL 1, PAGE 238)
C
DIMENSION XVAL(1),ISIG(1),AVE(10),SIG(10)
DIMENSION ECAVE(3000),ECSIG(3000)
IF(IB.NE.1) GO TO 2
DO 1 I=1,N
AVE(I) = XVAL(I)
ISIG(I)=99
1 SIG(I)=0.0
10 DO 100 I=1,N
IKK=K+I-1
ECAVE(IKK)=AVE(I)
ECSIG(IKK)=SIG(I)
K=K+N
RETURN
2 XNSORS=IB
DO 200 I=1,N
IKK=K+I-1
AVE(I)=ECAVE(IKK)
SIG(I)=ECSIG(IKK)
200 SIG(I)=ECSIG(IKK)
DO 3 I=1,N
SIG(I)=(XNSORS-2.0)/(XNSORS-1.0)*SIG(I)+(XVAL(I)-AVE(I))*2/XNSORS
AVE(I) = ((XNSORS-1.0)*AVE(I)+XVAL(I))/XNSORS
XVAL(I) = AVE(I)
ERROR=SQRT(SIG(I)/XNSORS)
ISIG(I)=99
IF(AVE(I).EQ.0.0) GO TO 3
ISIG(I)=100.0*ERROR/ABS(AVE(I))*50001
IF(ISIG(I).GT.99) ISIG(I)=99
3 CONTINUE
GO TO 10
END
SUBROUTINE ENERGY
C
C "ENERGY" PERFORMS A TABLE LOOK-UP ON THE ELECTRON ENERGY AND
C PROVIDES THE CORRESPONDING CROSS SECTION VALUE.
C
COMMON/MC/STO(7,51),NCOLL,CTMO,CTH,STM,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2AO,YO,ZC,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),AIRC
IF(EGY-.1)1,2,2
2 IE=1+IFIX(100.*(1.-EGY))
GO TO 3
1 IF(EGY.LT.0.01)EGY=.01
IE=91+IFIX(1000.*(1.-EGY))
3 DE=STAB(IE)*S
EGY=EGY-DE
IF(EGY.GT.ECUT)GO TO 4
IFLAG=1
EGY=ECUT
ENTRY CROSS
4 ETA=ETAB(IE)
AMBDA=1./SIGTAB(IE)
RETURN
END
SUBROUTINE ASSOC(PL,CX, SX,P)
C
C "ASSOC" COMPUTES THE ASSOCIATED LEGENDRE FUNCTIONS"

```

```

C
      SIGN PL(10),P(9,9)
      IF(JX.GT.0.0)GO TO 1
      DO 2 L=1,9
      DO 2 M=1,9
      2 P(L,M)=0.0
      RETURN
      1 P(1,1)=SX
      P(2,1)=3.*CX*SX
      P(2,2)=2.*CX/SX*P(2,1)-6.*PL(3)
      M=1
      LOW=M+1
      DO 10 L=LOW,8
      10 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L+M)*P(L-1,M))/FLOAT(L-M+1)
      P(3,2)=2.*CX/SX*P(3,1)-12.*PL(4)
      DO 100 M=2,8
      LOW=M+1
      IF(LOW.EQ.9)GO TO 51
      DO 50 L=LOW,8
      50 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L+M)*P(L-1,M))/FLOAT(L-M+1)
      51 P(LOW,M+1)=(2*M)*P(LOW,M)*CX/SX-((LOW+M)*P(LOW,M+1))*P(LOW,M-1)
      IF(M.EQ.8)GO TO 100
      P(LOW+1,M+1)=(2*M)*P(LOW+1,M)*CX/SX-((LOW+M+1)*P(LOW+M+1))*P(LOW+M+2))
      1P(LOW+1,M-1)
      100 CONTINUE
      RETURN
      END

```

Program Listing

- INTFC -

(Reference: Vol. I, Section IV.B.4)

```

OVERLAY(COMM,0.0)
PROGRAM INTFC(INPUT,OUTPUT,TAPE2,TAPE3)

THIS PROGRAM MAKES USE OF THE ELECTRON TRANSMISSION AND ALBEDO
DATA FROM "BIGSLAB" AND PERFORMS A DETAILED MONTE CARLO CALCULA-
TION IN THE REGION OF A TWO MATERIAL INTERFACE. IF THE ELECTRON
ESCAPES FROM THE INTERFACE REGION AND ENTERS THE THICK SLAB POR-
TION OF EITHER MATERIAL, THE ALBEDO DISTRIBUTION FROM "BIGSLAB" IS
SAMPLED SO THAT THE BACKSCATTERED ELECTRON IS POINTED BACK TOWARD
THE INTERFACE REGION WITH THE APPROPRIATE ENERGY AND DIRECTION.
THE TRANSMISSION DISTRIBUTION FROM "BIGSLAB" IS USED AS THE
INITIAL SOURCE FUNCTION INTO THE INTERFACE REGION. EXCEPT FOR
THESE FEATURES, THIS PROGRAM AND "BIGSLAB" ARE STRUCTURED
SIMILARLY. THE OUTPUT QUANTITIES ARE DEFINED IN VOL. I, PAGES
257-260.

COMMON/MC,STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHID,ED,SPAC,AMBD,A,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRNT,TLAST,CTHCH,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DATA CPHIO,SPHID,ZMIN,ZMAX,N/1.0,0.0,1000.,0/
DATA ECUT,SPAC,XO,YO,ZO/0.,800.,2*0.5-E-4/
DATA THICK/50.,51...52.,10000./
DATA WMIN/1.E-5/

THE FIRST OVERLAY IS THE MONTE CARLO CALCULATION

CALL OVERLAY(4HCOMM,1,0)

THE SECOND OVERLAY IS THE PROCESSING PROGRAM FOR THE TRAJECTORY
CHARACTERISTICS. THE OUTPUT QUANTITIES ARE COMPUTED HERE.

CALL OVERLAY(4HCOMM,2,0)
CALL EXIT
END
OVERLAY(COMM,1.0)
PROGRAM MCTR
CALL MCTR
END
SUBROUTINE MCTR
COMMON/MC,STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHID,ED,SPAC,AMBD,A,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRNT,TLAST,CTHCH,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DIMENSION TIMR(11)
CALL SETRUN
CALL SETRN
FN=FLOAT(NMAX)
COLNO=0.
MTN=1
CALL SECOND(TIME)
TIMR(1)=TIME
5 N=N+1
CALL SETHIS
CALL SCORE
CALL PENET
IF(IFLAG.EQ.1) GO TO 6
CALL ENERGY
IF(IFLAG.EQ.1) GO TO 6
20 SAMPLE THE ALBEDO DISTRIBUTION FROM "BIGSLAB"

```

```

IF (FLAG.EQ.7) CALL BACKS(EGY,CTH,STH,PHI,CPhi,SPHI,NF)
IF (LT.WMIN) GO TO 6
IF (LAG.NE.7) CALL ANGLES
IFLAG=0
GO TO 20
6 COLLNO=COLLNO+FLOAT(NCOLL)/FN
IF (NBATCH.EQ.0) GO TO 60
NTEST=N-NBATCH*(N/NBATCH)
IF (NTEST.NE.0) GO TO 60
MTM=MTM+1
CALL SECOND(TIME)
TIMR(MTM)=TIME
DELT=TIME-TIMR(1)
AVGT=DELT/FLOAT(MTM-1)
TREM=TIME-TIM-DELT
TLEFT=1.5*AVGT
IPCT=10*(MTM-1)
PRINT 75,IPCT,TREM
75 FORMAT(1X,'THE PROBLEM IS',14,' PERCENT COMPLETE.',1X,
1'TIME TO FINISH IS',F12.3,'SECONDS')
IF (IPCT.EQ.100) GO TO 100
IF (TREM.NE.0) GO TO 5
GO TO 100
60 IF (N-NMAX) 5,100,100
2 FORMAT(1X,'E16.9')
100 PRINT 64,COLLNO
IF (LCHR.GT.0) AND (LCHR.LE.201) WRITE(2)((STO(K,L),K=1,7),L=1,201)
IF (LCHR.EQ.0) WRITE(2)((STO(K,L),K=1,7),L=1,201)
CALL RANGET(XIRC)
64 FORMAT(1X,'AVERAGE NUMBER OF COLLISIONS PER HISTORY = ',F7.3)
IF (NBATCH.EQ.0) GO TO 200
MTM=MTM-1
PRINT 65,MTM,NBATCH
65 FORMAT(1X,'THERE ARE ',14,' BATCHES OF',15,' HISTORIES')
AVGT=0.
SUM=TIMR(1)
DO 66 M=2,MTM
TIMR(M)=TIMR(M)-SUM
SUM=SUM+TIMR(M)
AVGT=AVGT+TIMR(M)
MTM=MTM-1
66 PRINT 67,MTM,TIMR(M)
67 FORMAT(1X,'BATCH NO.',14,' TIME = ',F6.3,' SECONDS')
AVGT=AVGT/FLOAT(MTM)
PRINT 68,AVGT
68 FORMAT(1X,'AVERAGE TIME PER BATCH = ',F6.3,' SECONDS')
200 PRINT 190
190 FORMAT(1X,'INRAN',11X,'IRA',12X,'IRC')
PRINT 195,INRAN,XIRA,XIRC
195 FORMAT(3(1X,016))
RETURN
END
SUBROUTINE SETRUN
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPhi,SPHI,EGY,CPhiID,
1SPHIID,EGY,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,5,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DIMENSION INCDEL(3)
READ 1,NMAX,MAXCOLL,NPRNT,INPT,NSLABS
NMAX - TOTAL NO. OF MONTE CARLO HISTORIES
STORED ON TAPE OR DISK UNTIL PROCESSING TIME
MAXCOLL - MAXIMUM NUMBER OF COLLISIONS PER HISTORY TO BE ALLOWED
NPRNT - DEBUG PRINT CONTROL PARAMETER
NPRNT EQUAL TO ZERO SUPPRESSES DEBUG PRINTOUT
NPRNT NOT EQUAL TO ZERO ACTIVATES DEBUG PRINTOUT

```

```

READ 3, TLIM, CTHD, ED, ECUT
IF (LABS.EQ.0) GO TO 50
RE 3, (THICK(1), I=1, 11)
50 DO 51 I=1, 4
51 THICK(I)=THICK(1)+.1E-5
IF (INPT.GT.0) GO TO 40
INRAN=0
GO TO 45
40 READ 1, (NCODE(1), I=1, 3)
IF (NCODE(1).EQ.1) READ 8, INRAN
IF (NCODE(2).EQ.1) READ 3, SFAC
IF (NCODE(3).EQ.1) READ 3, XO, YO, ZO
ED - INITIAL PARTICLE ENERGY
XO, YO, ZO - INITIAL PARTICLE COORDINATES
SFAC - STOPPING POWER FUDGE FACTOR
45 PRINT 4, NMAX, MAXCOLL
PRINT 9, ED, ECUT, XO, YO, ZO, CTHD, CPHI, SPHIO
PRINT 11, SFAC
11 FORMAT(1X, 'STOPPING POWER FUDGE FACTOR (SFAC) = ', F10.5)
9 FORMAT(1X, 'ED (INITIAL ENERGY MEV) =', E12.5//1X, 'ECUT (LOW ENERGY
1 CUT OFF MEV) =', E12.5//1X, 'XO, YO, ZO, (SOURCE POINT COORDINATES)
3 =', E12.5//1X, 'CTHD (INCIDENT POLAR COSINE) =', E12.5//1X, 'CPHIO
4, SPHIO (INCIDENT AZIMUTH) =', E12.5)
8 FORMAT(D18)
1 FORMAT(G110)
3 FORMAT(8F10.0)
4 FORMAT(1X, 'NMAX (NO. OF HISTORIES) =', I6//1X, 'MAXCOLL (MAXIMUM
1 ALLOWED NUMBER OF COLLISIONS) =', I3)
10 N=0
LCNR=0
REWIND 2
XIRA=XIRC=INRAN
NBATCH=NMAX/10
CALL RANSET(XIRA)
CALL ESET
DO 20 K=1, 7
DO 20 L=1, 201
20 STO(K, L)=0.
DO 30 I=1, 10
DO 30 J=1, 101
CZ=.01*(I-1)
30 PLI(L, I)=PLINT(L, CZ)
RETURN
END
SUBROUTINE SETHIS
COMMON/MC/STO(7, 201), NCOLL, CTHD, CTH, STH, PHI, CPHI, SPHI, EGY, CPHIO,
1SPHIO, ED, SFAC, AMBDA, N, IFLAG, MAXCOLL, LCNR, STAB(200), ETAB(200), TLIM,
2XO, YO, ZO, X, Y, Z, ECUT, CLAB, THICK(11), NMAX, ZMIN, ZMAX, IE, INRAN, XIRA
3, NPRINT, TLAST, CTHC, NBATCH, ETA, PATH, S, SIGTAB(200), ENTAB(200), XIRC,
4PLI(10, 101), WF, WMIN
DIMENSION ESCR(9), PSRC(8), NG(8)
DATA ESCR/.4, .35, .3, .25, .2, .15, .1, .05, .01/
DATA PSRC/.023713, .13685, .18255, .18768, .13856, .11498, .075147, .1405
12/
DATA JS/1/
IF (N.GT.1) GO TO 1
NN=0
DO 2 JU=1, 8
NG(JU)=NMAX*PSRC(JU)
2 NN=NN+NG(JU)
NLEFT=NMAX-NN
NG(8)=NG(8)+NLEFT
DO 3 JU=2, 8
3 NG(JU)=NG(JU)+NG(JU-1)
PRINT 17, (NG(JU), JU=1, 8)
17 FORMAT(1X, 'CUMULATIVE TOTAL OF HISTORIES FOR THE 8 ENERGY GROUPS=')

```

```

1 IF (A(10))
1 PA 0.
NCOLL=0
S=0
X=XD
Y=YD
Z=ZD
WF=.32736
CPHI=CPHIO
SPHI=SPHIO
PHI=0.0
IF(N.GT.NG(JS))JS=JS+1
RA=RA*(10)
ED=ED*ESRC(JS)*ESRC(JS+1)*(1.-RA)
JSRC=(JS+1)/2
C C C
C SAMPLE THE SOURCE ENERGY ANGLE DISTRIBUTION FROM A "BIGSLAB" RUN
C C C
CALL SAMPMU('SRC')
CTH=CTHO
STH=STHT(1.-CTH**2)
EGY=EO
IF(EGY-.1)/7.8.8
8 IE=1+IFIX(100.*(1.-EGY))
GO TO 9
7 IF(EGY.LT..001)EGY=.001
IE=91+IFIX(1000.*(1.-EGY))
9 CALL CROSS
IFLAG=0
RETURN
END
SUBROUTINE GENET
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
X1=X
Y1=Y
Z1=Z
4 RB=RANF(0)
IF(RB.GT.0.)GO TO 5
GO TO 4
5 S=-ALOG(RB)*AMBDA
Z=Z1+S*CTH
NCOLL=NCOLL+1
IF(Z.GT.THICK(2).AND.Z1.LE.THICK(2))CALL SCORE
IF(Z.LE.THICK(2).AND.Z1.GT.THICK(2))CALL SCORE
IF(Z.GE.THICK(1).AND.Z.LE.THICK(3))GO TO 10
IFLAG=7
IF(CTH)/6.6.7
6 IF(Z.LT.THICK(1))Z=THICK(1)
GO TO 8
7 IF(Z.GT.THICK(3))Z=THICK(3)
8 IF(CTH.NE.0.0)GO TO 88
S=1.E20
GO TO 10
88 S=(Z-Z1)/CTH
10 X=X1+S*STH*CPHI
Y=Y1+S*STH*SPHI
PATH=PATH+S
IF(NCOLL.EQ.MAXCOLL) IFLAG=1
RETURN
END
SUBROUTINE ESET
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,

```

```

158 1,ED,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2X0,3,ED,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,RA
3,NPRT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DATA PI,XL2/3.14159265,.593147181/
DATA ZA,AA,AVOG,RO,DNSTY,EXCIT,FCC/13,.26,98,6.025E23,2.819E-13,
12.7,163,E-6,.15360628/
ENTAB(1)=1.0
DE=.01
DO 1 11=2,91
1 ENTAB(11)=ENTAB(11-1)-DE
DE=.001
DO 2 11=92,190
2 ENTAB(11)=ENTAB(11-1)-DE
DO 3 11=1,190
E=ENTAB(11)/.511
E1=(1.+E)*(1.+E)
E2=E*(2.+E)
BETA=SQRT(E2/E1)
XLG=ALOG(E2/(2.*(EXCIT/.511)**2))
FM=1.-BETA*BETA*(.125+E-(2.+E+1.)*XL2)/E1
STAB(11)=(FCC-ZA/AA/BETA**2)*(XLG-FM)*SFAC
ETAB(11)=0.5*(ZA**((1./3.)/121.245)**2/E2*(1.13+3.76*(ZA/137.))**2*
1E1/E2)
SIGTAB(11)=ZA*(1.+ZA)*AVOG*RO*RO*ZA/AA*E1/(E2*E2)/(ETAB(11)
1*(1.+S*ETAB(11)))
ETAB(11)=.035276
3 IF(11.GT.91)ETAB(11)=.103378
IF(NPRT.EQ.0)RETURN
PRINT 75
75 FORMAT(1X,'CROSS SECTION AND STOPPING POWER TABLES',//,1X,'ENERGY(M
1EV) STOPPING POWER ETA SIGMA')
DO 76 11=1,190
76 PRINT 77,ENTAB(11),STAB(11),ETAB(11),SIGTAB(11)
77 FORMAT(1X,F10.6,3E16.9)
RETURN
END
SUBROUTINE ANGLES
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIQ,
1SPHIQ,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2X0,Y0,Z0,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DATA TWOPI/6.283185307/
RD=2.*RANF(0)
COM=(RD*(1.+ETA)-ETA)/(RD+ETA)
SOM=SQRT(1.-COM**2)
RC=RANF(0)
RHO=TWOPI*RC
CRHO=COS(RHO)
SRHO=SIN(RHO)
STH=SIN(RHO)
CTH=CTH
CTH=CTH*COM+STH*SOM*CRHO
STH=SQRT(1.-CTH**2)
IF(STH1.EQ.0.0.OR.STH.EQ.0.0) GO TO 1
C1=(COM-CTH*CTH1)/(STH*STH1)
S1=SRHO*SOM/STH
CPHI1=CPHI
SPHI1=SPHI
CPHI=CPHI1+C1*SPHI1*S1
SPHI=CPHI1+S1*SPHI1*C1
GO TO 2
1 CPHI=CRHO
SPHI=SRHO
2 CONTINUE

```



```

PH=ATAN2(SPHI,CPHI)
I=11.LT.0.0)PHI=PHI+TWOP1
RE=JRN
END
SUBROUTINE LEP(Y,X,N)
  DIMENSION Y(1)
  Y(1)=1.
  IF(N1).1.2
  1 RETURN
  2 Y(2)=X
  IF(N-1).1.3
  3 DO 4 I=2,N
    G=X*Y(I)
  4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
  RETURN
END
SUBROUTINE ENERGY
COMMON/MC/STOI7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
  IF(EGV-1).1.2.2
  2 IE=1+FIX(100.*(1.-EGY))
  GO TO 3
  1 IF(EGY.LT.0.001)EGY=.001
  IE=91+FIX(1000.*(1.-EGY))
  3 DE=STAB(IE)*S
  EGY=EGY-DE
  IF(EGY.GT.ECUT)GO TO 4
  IFLAG=1
  EGY=ECUT
  ENTRY CROSS
  4 ETA=ETAB(IE)
  AMBDA=1./SIGTAB(IE)
  RETURN
END
SUBROUTINE ASSOQ(PL,CX,SX,P)
  COMPUTE ASSOCIATED LEGENDRE FUNCTIONS
  DIMENSION PL(10),P(9,9)
  IF(SX.GT.0.0)GO TO 1
  DO 2 L=1,9
    DO 2 M=1,9
      P(L,M)=0.0
  RETURN
  1 P(1,1)=SX
  P(2,1)=3.*CX*SX
  P(2,2)=2.*CX/SX+P(2,1)-8.*PL(3)
  M=1
  LOW=M+1
  DO 10 L=LOW,8
    P(L+1,M)=((2*L+1)*CX*P(L,M)-(L*N)*P(L-1,M))/FLOAT(L-M+1)
    P(3,2)=2.*CX/SX+P(3,1)-12.*PL(4)
    DO 100 M=2,8
      LOW=M+1
      IF(LOW.EQ.9)GO TO 51
      DO 50 L=LOW,8
        P(L+1,M)=((2*L+1)*CX*P(L,M)-(L*N)*P(L-1,M))/FLOAT(L-M+1)
        51 P(L,M+1)=((2*M)*P(L,M)+CX/SX-((LOW+M)*(LOW-M+1))*P(LOW,M-1)
          IF(M.EQ.8)GO TO 100
          P(LOW+1,M+1)=(2*M)*P(LOW+1,M)+CX/SX-((LOW+M+1)*(LOW-M+2))*
            P(LOW+1,M-1)
        100 CONTINUE
      RETURN

```

```

END
SUP  UTINE SAMPNU(JSRC)
C
C THIS ROUTINE SAMPLES THE TRANSMISSION DISTRIBUTION DATA FROM A
C "BIGSLAB" RUN
C
COMMON/NC/STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGV,CPHIO,
15PHIO,EQ,EFAC,AMBDA,N,IFLAG,MAXCOLL,LCRR,STAB(200),ETAB(200),ILIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DIMENSION LMAX(4),PCUM(101),TLEG(10,4)
DATA TLEG/.14827,.22613,.22719,.18634,.13199,.083137,.047188,.0240
163,.01107,.0052421,.34190,.47020,.3737,.19678,.054955,-.012527,
2-.029918,-.03217,-.033122,-.030991,.23414,.29725,.19183,.055359,
3-.01838,-.024905,-.0090189,-.002056,.0031651,.0089282,
4.19916,.2388,.12831,.0085602,-.033786,-.013766,.0091091,
5.008585,.00045677,.00098932/
DATA JOLD/O/
DATA LMAX/8,7,7,7/
IF(JSRC.EQ.JOLD)GO TO 10
JOLD=JSRC
LM=LMAX(JSRC)
DO 1 K=1,101
PCUM(K)=0.
DO 1 L=1,LM
AL=L
FAC=SQRT(2.*AL-1.)
PCUM(K)=PCUM(K)+FAC*TLEG(L,JSRC)*PLI(L,K)
1 CONTINUE
DO 2 K=1,101
IF(PCUM(K).LT.O.)PCUM(K)=0.
2 PCUM(K)=PCUM(K)/PCUM(101)
PRINT 3,JSRC,(PCUM(K),K=1,101)
3 FORMAT(1X,'SOURCE GROUP NO.',12/
11X,'CUMULATIVE PROBABILITIES OF EMERGENT SOURCE ANGLE(0.-1...01)')
2(1X,10E12.5))
10 RA=RAHF(O)
DO 11 K=1,101
IF(RA-PCUM(K))12,12,11
11 CONTINUE
K=101
12 CTHO=.01*(K-1)-.01*(PCUM(K)-RA)/(PCUM(K)-PCUM(K-1))
RETURN
END
FUNCTION PLINT(L,X)
C
C COMPUTE INTEGRALS OF LEGENDRE POLYNOMIALS TO ORDER 10.
C
Y=X*X
Z=Y*Y
GO TO (1,2,3,4,5,6,7,8,9,10),L
1 PLINT=X
RETURN
2 PLINT=0.5*Y
RETURN
3 PLINT=0.5*X*(Y-1.)
RETURN
4 PLINT=0.25*Y*(2.5*Y-3.)
RETURN

```

```

5 PLINT=0.125*X*(7.*Z-10.*Y+3.)
6 PL(N=0.0625*Y*(21.*Z-35.*Y+15.))
7 RETURN
8 PLINT=0.0625*X*(33.*Y+Z-63.*Z+35.*Y-5.)
9 RETURN
10 PLINT=0.0625*(429./8.*Z+Z-231./2.*Z+Y+315./4.*Z-35./2.*Y)
11 RETURN
12 PLINT=X/128.*(715.*Z+Z-1716.*Y+Z+1386.*Z-420.*Y+35.)
13 RETURN
14 PLINT=Y/128.*(2431./2.*Z+Z-6435./2.*Z+Y+3003.*Z-1155.*Y+315./2.)
15 RETURN
16 END
17 SUBROUTINE SCORE
18 COMMON/MC/STO(7,201),NCOLL,CTHD,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIG,
19 1SPHIG,ED,EFAC,AMSDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
20 2XO,YO,ZO,X,Y,Z,ECUT,CNLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
21 3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
22 4PLI(10,101),WF,WMIN
23 70 LCHR=LCHR+1
24 STO(1,LCHR)=1.0
25 IF(NCOLL.EQ.0)STO(1,LCHR)=100.
26 STO(2,LCHR)=CTHD
27 STO(3,LCHR)=CTH
28 STO(4,LCHR)=STH
29 STO(5,LCHR)=PHI
30 STO(6,LCHR)=EGY
31 STO(7,LCHR)=WF
32 IF(LCHR.EQ.201)GO TO 95
33 WRITE(2)((STO(MM,LL),MM=1,7),LL=1,LCHR)
34 IF(NPRINT.EQ.2)PRINT 2,((STO(MM,LL),MM=1,7),LL=1,LCHR)
35 2 FORMAT(1X,7E16.9)
36 IF(LCHR.EQ.0)WRITE(2)((STO(K,L),K=1,7),L=1,201)
37 DO 1 MM=1,7
38 DO 1 LL=1,LCHR
39 1 STO(MM,LL)=0.
40 LCHR=0
41 95 IF(NCOLL.EQ.0)RETURN
42 100 CONTINUE
43 RETURN
44 END
45 SUBROUTINE BACKS(EGY,CTH,STH,PHI,CPHI,SPHI,WF)
46 SAMPLE ALBEDO DISTRIBUTIONS FROM "BIGSLAB".
47 DIMENSION AZERO(30,10,10),AP(9),PTEST(10),
48 1ETOP(5),AM(9),FL(9),EDUT(11,5),PCUM(101),FAC(10),
49 2LMAX(10,6,5),TOTALB(5,6),EPROB(5,6,11),YCOF(9,9),PLM(9,9),
50 3CPI(9,11),SPI(9,11),PPH(11),PL(10),PPHCUM(11),PLI(10,101)
51 DIMENSION APLUS(30,10,45),AMINUS(30,10,45)
52 REAL MU(6),RATIO(10)
53 DATA IEQ,ITIME/0,0/
54 DATA MU/1.,.9,.7,.5,.3,.1/
55 DATA ETOP/4.,3.,2.,1.,.05/
56 DATA SOFPI,PI/3.544907702,3.141592654/
57 DATA TMOP1/6.283185307/
58 IF(ITIME.EQ.1)GO TO 100
59 IWO=1
60 1 IEO=IEO+1
61 REWIND 3
62 ITIME=1
63 IF(IEQ.GT.1)IWO=IEO-1
64 DO 10 IWO=1,6
65 IO=6*(IWO-1)+IWO
66 DO 9 KE=1,10
67 READ(3)(AZERO(IO,KE,L),L=1,10)

```

```

      LM=L-1.9
      DO 10 L=1,9
        READ(3)(AP(M),M=1,L)
        READ(3)(AM(M),M=1,L)
        DO 7 M=1,L
          LM=LM+1
          APLUS(10,KE,LM)=AP(M)
          7 AMINUS(10,KE,LM)=AM(M)
          8 CONTINUE
          9 CONTINUE
          10 CONTINUE
          IF(IEO.LT.6)GO TO 1
          PPH(1)=0.
          DO 16 L=1,10
            DO 34 J=1,101
              CZ=.01*(J-1)
              34 PLI(L,J)=PLINT(L,CZ)
              AL=L
              16 FAC(L)=SORT(2.*AL-1.)
              DFHI=PI/10.
              DO 29 KP=2,11
                PPH(KP)=PPH(KP-1)+DPHI
              29 PPH(KP)=PPH(KP-1)+DPHI
              DO 30 L=1,9
                FL(L)=(2.*FLOAT(L)+1.)/6.283185307
                DO 31 M=1,L
                  LM=L-M+1
                  LM=L+M
                  FI=1.
                  DO 32 I=LM,LM
                    FI=FI+FLOAT(I)
                    32 FI=FI+FLOAT(I)
                    31 YCOF(L,M)=SQRT(FL(L)/FI)
                    DO 33 KP=1,11
                      CP(L,KP)=COS(L*PPH(KP))
                      33 SP(L,KP)=SIN(L*PPH(KP))
                    30 CONTINUE
                    EPROB IS THE CUMULATIVE PROBABILITY OF EXIT WITH ENERGY E
                    C
                    C
                    GIVEN EO,MUO
                    DO 5 IEO=1,5
                      EOUT(1,IEO)=ETOP(IEO)
                      DE=ETOP(IEO)/10.
                      DO 4 KE=2,11
                        4 EOUT(KE,IEO)=EOUT(KE-1,IEO)-DE
                        DO 3 IMO=1,6
                          TOTALB(IEO,IMO)=0.
                          IO=6*(IEO-1)+IMO
                          DO 2 KE=1,10
                            2 TOTALB(IEO,IMO)=TOTALB(IEO,IMO)+SQFPI*AZERO(10,KE,1)
                            EPROB(IEO,IMO,1)=0.
                            DO 3 KE=2,11
                              EPROB(IEO,IMO,KE)=SQFPI*AZERO(10,KE-1,1)/TOTALB(IEO,IMO)
                            3 CONTINUE
                            5 CONTINUE
                            DO 11 IEO=1,5
                              DO 11 IMO=1,6
                                DO 11 KE=2,11
                                  11 EPROB(IEO,IMO,KE)=EPROB(IEO,IMO,KE)+EPROB(IEO,IMO,KE-1)
                                  PRINT 60,IEO,IMO
                                  IO=6*(IEO-1)+IMO
                                  DO 40 KE=1,10
                                    IF(AZERO(10,KE,1).EQ.0.0)GO TO 40
                                    PC=0.
                                    DO 41 L=1,10
                                      AL=L
                                      FC-SORT(2.*AL-1.)/SQFPI*2.*PI

```

AD-A139 684

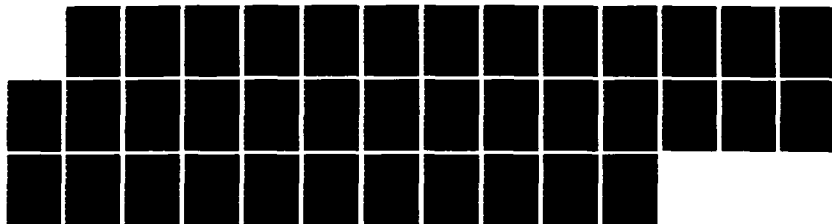
MATHEMATICAL MODELLING OF WAVEGUIDING TECHNIQUES AND
ELECTRON TRANSPORT VOLUME 2(U) ARCON CORP WALTHAM MA
S WOOLF ET AL. JAN 84 RADC-TR-83-313-VOL-2
F19628-78-C-0188

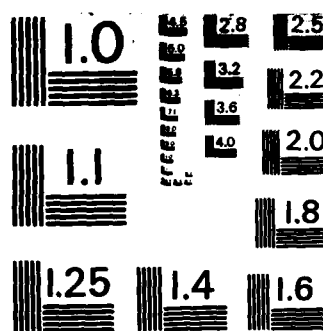
5/5

UNCLASSIFIED

F/G 9/2

NL





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

```

PQ 7-FC-AZERO(10,KE,L)*PLI(L,101)
41 P I(L)=PC
ATEST-SOPFI-AZERO(10,KE,1)
DIFF=1.E6
DO 42 L=1,10
RATIO(L)=(PTEST(L)/AZERO(10,KE,1))*2/(4.*PI)
DD=RATIO(L)-1.
IF(DD)42,47,48
47 LOW=L
GO TO 49
48 LOW=L
AD=1.-RATIO(L-1)
IF(AD.LT.DD)LOW=L-1
GO TO 49
42 CONTINUE
LOW=10
49 IF(LOW.EQ.10)GO TO 491
IF(LOW.NE.(L-1))GO TO 491
IF(RATIO(LOW+1).EQ.RATIO(LOW))LOW=LOW+1
491 CONTINUE
LMAX(KE,IMO,IED)=LOW
40 CONTINUE
50 CONTINUE
DO 58 IEO=1,5
DO 58 IMO=1,6
PRINT 57,IEO,IMO
58 FORMAT(1X,1015)
57 FORMAT(1X,IEO=.,IS,1X,IMO=.,IS)
60 FORMAT(1X,IEO=.,IS,2X,IMO=.,IS)
C CLASSIFY INCIDENT ENERGY AND ANGLE
C FIND CLOSEST VALUES IN TABLE
100 DTEST=100.
DO 101 II=1,5
DE=ABS(EGY-ETOP(II))
IF(DE.GT.DTEST)GO TO 101
IEO=II
DTEST=DE
101 CONTINUE
DTEST=100.
CT=ABS(CTH)
DO 102 II=1,6
DD=ABS(CT-MUO(II))
IF(DD.GT.DTEST)GO TO 102
IMO=II
DTEST=DD
102 CONTINUE
C WEIGHT FACTOR
WF=TOTALB(IEO,IMO)*WF
C COMPUTE NEW ENERGY
590 RA=RAHF(0)
DO 103 KE=2,11
IF(RA-EPROB(IEO,IMO,KE))104,104,103
103 CONTINUE
KE=11
104 DELTA=(EOUT(KE,IED)-EOUT(KE-1,IED))/(EPROB(IEO,IMO,KE)-
1 ETST-EOUT(KE,IED)-EPROB(IEO,IMO,KE-1))
IF(ETST-GE.EGY)GO TO 590
EGY=ETST
C COMPUTE EXIT POLAR COSINE MU FROM LEGENDRE COEFFICIENTS
KEM=KE-1
LMAX=LMAX(KEM,IMO,IED)
DO 105 I=1,101
105 PCUM(I)=0.
DO 107 I=2,101

```

```

DO 106 LL=1,LWX
106 PC(I)=PCUM(I)*FAC(LL)*PLI(LL,I)*AZERO(IO,KEM,LL)
107 CONTINUE
DO 108 I=1,101
IF(PCUM(I).LT.0.)PCUM(I)=0.
PCUM(I)=PCUM(I)/PCUM(101)
108 IF(PCUM(I).GT.1.0)PCUM(I)=1.0
RA=RA*F(0)
DO 109 I=1,101
IF(RA-PCUM(I))110,110,109
109 CONTINUE
110 SI=1.
IF(CTH.GT.0.)SI=-1.
CTH=SI*(1.01*(I-1)-.01*(PCUM(I)-RA)/(PCUM(I)-PCUM(I-1)))
DETERMINE EXIT AZIMUTH
STH=SQRT(1.-CTH*CTH)
AC=ABS(CTH)
CALL LEP(PL,AC,9)
CALL ASSOC(PL,AC,STH,PLM)
254 SUM1=0.
DO 201 L=1,LWX
201 SUM1=SUM1+FAC(L)/SOFFI*AZERO(IO,KEM,L)*PL(L)
LWX1=LWX-1
SUM2=0.
DO 202 KP=1,11
A1=PPH(KP)*SUM1
SUM3=0.
DO 203 L=1,LWX1
SUM2=0.
DO 204 M=1,L
LM=(L*(L+1))/2-(L-M)
EN=M
204 SUM2=SUM2+YCOF(L,M)/EM*PLM(L,M)*
1(APLUS(IO,KEM,LM)*SP(M,KP)-AMINUS(IO,KEM,LM)*(CP(M,KP)-1.))
SUM3=SUM3+SUM2
203 CONTINUE
PPHCUM(KP)=A1+SUM3
202 CONTINUE
DO 255 KP=2,11
IF(PPHCUM(KP).LT.PPHCUM(KP-1))GO TO 256
255 CONTINUE
GO TO 253
256 LMX=LWX-1
IF(LMX.EQ.1)GO TO 257
GO TO 254
257 DO 258 KP=2,11
258 PPHCUM(KP)=PPHCUM(KP-1)+.1
253 DO 205 KP=1,11
205 PPHCUM(KP)=PPHCUM(KP)/PPHCUM(11)
RA=RA*F(0)
DO 206 KP=1,11
IF(RA-PPHCUM(KP))208,208,206
206 CONTINUE
KP=11
208 PPH=PPH(KP)-DPHI*(PPHCUM(KP)-RA)/(PPHCUM(KP)-PPHCUM(KP-1))
RA=RA*F(0)
IF(RA.GE.0.5)PHP=TWOP1-PHP
PHI=PHI+PHP
IF(PHI.GT.TWOP1)PHI=PHI-TWOP1
CPHI=COS(PHI)
SPHI=SIN(PHI)
RETURN
END
OVERLAY(COMM,2.0)
PROGRAM PROX

```



```

CALL PROC
END
SUB JTINE PROC
COMMON/NC/STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,C,IO,
1SPHIO,EO,SPAC,AMBDA,N,IPLAG,MAXCOLL,LCHRR,STAB(200),ETAB(200),TLM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4PLI(10,101),WF,WMIN
DIMENSION TRANC(10,10),TRANF(10,10),BACKC(10,10),BACKF(10,10),
1APLUS(9,9,10),AMINUS(9,9,10),TPLUS(9,9,10),TMINUS(9,9,10),
2ALEG(10,10),TLEG(10,10),ENERGY(10),YCOF(9,9),ANGLE(10),PL(10),
3PLN(9,9),CP(9),SP(9),SUM(10),TEMP(10),ISIG(10),ITP(9,9,10),
4ITM(9,9,10),ISGEX(10,10),IPR(10)
DO 1 L=1,9
FL=(2.-FLOAT(L)+1.)/6.283185307
DO 2 M=1,L
LM1=L-M+1
LM=L-M
FI=1.
DO 3 I=LMI,LM
3 FI=FI-FLOAT(I)
2 YCOF(L,M)=SQRT(FL/FI)
1 CONTINUE
REWIND 2
NN=NMAX/10
FN=FLOAT(NNN)
EO=0.-4
DE=EO/10.
ENERGY(1)=EO-DE
ANGLE(1)=.1
DO 4 K=2,10
ANGLE(K)=ANGLE(K-1)+.1
4 ENERGY(K)=ENERGY(K-1)-DE
IPLANE=1
100 REWIND 2
IB=NH=NHIST=0
IPLANE=IPLANE+1
PRINT 777,THICK(IPLANE)
777 FORMAT('1=,40X,0DETECTOR PLANE AT=,E12.5)
DO 40 I=1,10
IPR(I)=0
DO 41 J=1,10
TRANC(I,J)=TRANF(I,J)=BACKC(I,J)=BACKF(I,J)=0.
41 ALEG(I,J)=TLEG(I,J)=0.
DO 40 J=1,9
DO 40 K=1,9
40 APLUS(J,K,I)=AMINUS(J,K,I)=TPLUS(J,K,I)=TMINUS(J,K,I)=0.
5000 READ(2)((STO(K,L),K=1,7),L=1,201)
DO 90 L=1,201
IF(STO(L,L).EQ.0.)GO TO 11
IF(STO(L,L).LT.10.)GO TO 10
NH=NH+1
11 NHIST=NHIST+1
GO TO 50
10 W=STOI(7,L)
CLASSIFY EMERGENT ANGLE
CA=ABS(STOI(3,L))
DO 80 J=1,10
80 IF(CA.LT.ANGLE(J))GO TO 81
J=10
81 JC=J
C CLASSIFY EMERGENT ENERGY
DO 82 K=1,10
82 IF(STO(6,L).GT.ENERGY(K))GO TO 83
K=10
83 KC=K

```

```

C
1PR(KC)=1
C/ POSITIVE OR NEGATIVE?
CH( -STD(3,1)
STH(=STO(4,1)
PHI=STO(5,1)
CALL LEP(PL,CTHET,9)
DO 70 M=1,9
CP(M)=COS(M*PHI)
CALL ASSOC(PL,CTHET,STHET,PLM)
IF(CTHET)84,85,85
84 BACKC(JC,KC)=BACKC(JC,KC)*W/FN
BACKF(JC,KC)=BACKF(JC,KC)-W/CTHET/FN
DO 86 LL=1,10
86 ALEG(LL,KC)=ALEG(LL,KC)+PL(LL)*W/FN
DO 87 LL=1,9
DO 88 M=1,LL
FM=YCOF(LL,M)*PLM(LL,M)/FN
APLUS(LL,M,KC)=APLUS(LL,M,KC)+FM*CP(M)*W
89 AMINUS(LL,M,KC)=AMINUS(LL,M,KC)+FM*SP(M)*W
87 CONTINUE
GO TO 90
85 TRANC(JC,KC)=TRANC(JC,KC)*W/FN
TRANF(JC,KC)=TRANF(JC,KC)+W/FN/CTHET
DO 89 LL=1,10
89 TLEG(LL,KC)=TLEG(LL,KC)+PL(LL)*W/FN
DO 91 M=1,LL
FM=YCOF(LL,M)*PLM(LL,M)/FN
TPLUS(LL,M,KC)=TPLUS(LL,M,KC)+FM*CP(M)*W
91 TMINUS(LL,M,KC)=TMINUS(LL,M,KC)+FM*SP(M)*W
99 CONTINUE
GO TO 90
50 IF(NHIST.LE.NMN)GO TO 90
NHIST=1
KPUT=1
18=18+1
DO 150 NM=1,4
DO 164 I=1,10
164 SUM(I)=0.
IF(18.LT.10)GO TO 154
PRINT 220
GO TO (114,124,134,144),NM
114 PRINT 230
GO TO 153
124 PRINT 231
GO TO 153
134 PRINT 232
GO TO 153
144 PRINT 233
153 PRINT 234,(ANGLE(I),I=1,10)
230 FORMAT(///45X,*,POSITIVE CURRENT*//)
231 FORMAT(///45X,*,NEGATIVE CURRENT*//)
232 FORMAT(///45X,*,POSITIVE FLUX*//)
233 FORMAT(///45X,*,NEGATIVE FLUX*//)
234 FORMAT(45X,*,EMERGENT POLAR COSINE*/9X,10E12.5/3X,*,E(MEV)*//)
154 DO 104 N=1,10
DO 105 I=1,10
GO TO (115,125,135,145),NM
115 TEMP(I)=TRANC(I,K)
TRANC(I,K)=0.
GO TO 105
125 TEMP(I)=BACKC(I,K)
BACKC(I,K)=0.
GO TO 105
135 TEMP(I)=TRANF(I,K)

```



```

1925 C      PRINT 524,ENERGY(M),(190EX(LL,M),LL=1,10)
      DO 345 M=1,10
      IF(1PR(ME).EQ.0)GO TO 1345
      PRINT 1347,ENERGY(ME)
1347 FORMAT(/1X,'ENERGY = ',F10.5,' MEV'/1X,'SPHERICAL HARMONICS COEFF
      ICIENTS FOR TRANSMISSION, TPLUS')
      DO 1343 LL=1,9
1343 PRINT 345,LL,(TPLUS(LL,M,KE),M=1,LL)
      PRINT 651
      DO 1652 LL=1,9
1652 PRINT 653,LL,(1TP(LL,M,KE),M=1,LL)
      PRINT 1348
1348 FORMAT(1X,'SPHERICAL HARMONICS COEFFICIENTS FOR TRANSMISSION, TMIN
      IUS')
      DO 1342 LL=1,9
1342 PRINT 349,LL,(TMINUS(LL,M,KE),M=1,LL)
      PRINT 651
      DO 1654 LL=1,9
1654 PRINT 653,LL,(1TM(LL,M,KE),M=1,LL)
1345 CONTINUE
1730 DO 700 I=1,9
      DO 700 J=1,9
      DO 701 K=1,10
      TEMP(K)=APLUS(I,J,K)
701 APLUS(I,J,K)=0
      CALL STATS(TEMP,ISIG,10,KPUT,IB)
      IF(1B.LT.10)GO TO 700
      DO 702 K=1,10
      APLUS(I,J,K)=TEMP(K)
702 ITP(I,J,K)=ISIG(K)
700 CONTINUE
      DO 710 I=1,9
      DO 710 J=1,9
      DO 711 K=1,10
      TEMP(K)=AMINUS(I,J,K)
711 AMINUS(I,J,K)=0
      CALL STATS(TEMP,ISIG,10,KPUT,IB)
      IF(1B.LT.10)GO TO 710
      DO 712 K=1,10
      AMINUS(I,J,K)=TEMP(K)
712 ITM(I,J,K)=ISIG(K)
710 CONTINUE
      DO 720 I=1,10
      DO 721 J=1,10
      TEMP(J)=ALEG(I,J)
721 ALEG(I,J)=0
      CALL STATS(TEMP,ISIG,10,KPUT,IB)
      IF(1B.LT.10)GO TO 720
      DO 722 J=1,10
      ALEG(I,J)=TEMP(J)
722 ISGEX(I,J)=ISIG(J)
720 CONTINUE
      IF(1B.LT.10)GO TO 90
C      PRINT LEGENDRE COEFFICIENTS
523 FORMAT(/45X,'LEGENDRE COEFFICIENTS OF BACKSCATTER',/4X,'E(MEV) L=
      1',/4X,'0',/1X,'1',/1X,'2',/1X,'3',/1X,'4',/1X,'5',/1X,'6',/1X,'7',
      /1X,'8',/1X,'9')
      DO 625 LL=1,10
      AL=LL
      FAL=SQRT((2.-AL-1.)/12.56837061)
      DO 625 M=1,10
      ALEG(LL,M)=ALEG(LL,K)*FAL
625 CONTINUE
      PRINT 523
      DO 626 M=1,10

```

```

1 (PR(K).EQ.0)GO TO 628
PR..T 526,ENERGY(M),(ALEG(LL,K),LL=1,10)
628 CONTINUE
526 FORMAT(3X,F7.3,1X,10E12.5)
PRINT 651
DO 525 K=1,10
IF(IPR(K).EQ.0)GO TO 525
PRINT 524,ENERGY(K),(ISGE(LL,K),LL=1,10)
525 CONTINUE
524 FORMAT(3X,F7.3,1X,10I4)
DO 345 K=1,10
IF(IPR(K).EQ.0)GO TO 345
3461 PRINT 347,ENERGY(K)
347 FORMAT(/1X,ENERGY = ,F10.5,SPHERICAL HARMONICS COEFF
1CIENTS - APLUS/)
DO 343 LL=1,9
343 PRINT 349,LL,(APLUS(LL,M,KE),M=1,LL)
349 FORMAT(1X,L=,15,1X,9E12.5)
PRINT 651
651 FORMAT(1X,PERCENT ERRORS IN ABOVE ITEMS=)
DO 652 LL=1,9
652 PRINT 653,LL,(ITP(LL,M,KE),M=1,LL)
PRINT 348
348 FORMAT(1X,SPHERICAL HARMONICS COEFFICIENTS - AMINUS=)
DO 342 LL=1,9
342 PRINT 349,LL,(AMINUS(LL,M,KE),M=1,LL)
PRINT 651
DO 654 LL=1,9
654 PRINT 653,LL,(ITM(LL,M,KE),M=1,LL)
345 CONTINUE
653 FORMAT(1X,L=,15,1X,9I4)
GO TO 200
90 CONTINUE
GO TO 5000
200 RETURN
END
SUBROUTINE LEP(Y,X,N)
DIMENSION Y(1)
Y(1)=1.
IF(N)1,1.2
1 RETURN
2 Y(2)=X
IF(N-1)1,1.3
3 DO 4 I=2,N
G=X*Y(I)
4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
RETURN
END
SUBROUTINE STATS(AVAL,ISIG,N,K,IB)
DIMENSION XVAL(1),ISIG(1),AVE(10),SIG(10)
DIMENSION ECAGE(4200),ECSIG(4200)
IF(18.NE.1) GO TO 2
DO 1 I=1,N
AVE(I) = XVAL(I)
ISIG(I)=99
1 SIG(I)=0.0
10 DO 100 I=1,N
IKK=K+I-1
ECAGE(IKK)=AVE(I)
100 ECSIG(IKK)=SIG(I)
K=K+N
RETURN
2 XNSORS=18
DO 200 I=1,N
IKK=K+I-1
AVE(I)=ECAGE(IKK)

```

```

200 SIG(I)=ECSIG(IKK)
DO 1 I=1,N
  SIG(I)=(XNSORS-2.0)/(XNSORS-1.0)*SIG(I)+(XVAL(I)-AVE(I))*2.0/XNSORS
  AVE(I) = ((XNSORS-1.0)*AVE(I)+XVAL(I))/XNSORS
  XVAL(I) = AVE(I)
  ERROR=SQRT(SIG(I)/XNSORS)
  ISIG(I)=99
  IF(AVE(I).EQ.0.0) GO TO 3
  ISIG(I)=100.0*ERROR/ABS(AVE(I))+.50001
  IF(ISIG(I).GT.99) ISIG(I)=99
3 CONTINUE
GO TO 10
END
SUBROUTINE ASSOC(PL,CX,SX,P)
  DIMENSION PL(10),P(9,9)
  IF(SX.GT.0.0)GO TO 1
  DO 2 L=1,9
    DO 2 M=1,9
      2 P(L,M)=0.0
  RETURN
  1 P(1,1)=SX
  P(2,1)=3.*CX*SX
  P(2,2)=2.*CX/SX*P(2,1)-6.*PL(3)
  M=1
  LOW=M+1
  DO 10 L=LOW,8
    10 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L+M)*P(L-1,M))/FLOAT(L-M+1)
    P(3,2)=2.*CX/SX*P(3,1)-12.*PL(4)
    DO 100 M=2,8
      LOW=M+1
      IF(LOW.EQ.9)GO TO 51
      DO 50 L=LOW,8
        50 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L+M)*P(L-1,M))/FLOAT(L-M+1)
        51 P(LOW,M+1)=(2*M)*P(LOW,M)*CX/SX-((LOW+M)*P(LOW,M+1)+P(LOW,M-1)
          IF(M.EQ.8)GO TO 100
          P(LOW+1,M+1)=(2*M)*P(LOW+1,M)*CX/SX-((LOW+M+1)*P(LOW+M+1)+P(LOW+M-1,M-1)
            1P(LOW+1,M-1)
        100 CONTINUE
      RETURN
    END

```

Program Listing

- MCSPLIT -

(Reference: Vol. I, Section IV.B.4)

```

PROGRAM MCSPLIT(INPUT,OUTPUT,TAPE2)

THE PURPOSE OF THIS PROGRAM IS TO SERVE AS A TEST OF THE
APPLICATION OF THE DISTRIBUTIONS COMPUTED BY "BIGSLAB" TO THE
INTERFACE CODE "INTFC". THE INTENTION IS TO USE THIS CODE
"MCSPILT" AS A BENCHMARK PROGRAM. IT MAKES USE OF THE STATISTICAL
BIASING TECHNIQUE KNOWN AS HISTORY SPLITTING(SEE REF.3, VOL. 1,
PAGE 264). IT IS A TECHNIQUE COMMONLY USED IN DEEP PENETRATION
MONTE CARLO CALCULATIONS. "MCSPILT" IS A STRAIGHTFORWARD SINGLE
SCATTERING ANALOG MONTE CARLO PROGRAM FOR CALCULATING ELECTRON
TRANSPORT IN A VERY THICK MEDIUM (GOLD-ALUMINUM) WITH THE MATERIAL
INTERFACE DEEPLY IMBEDDED. THE ELECTRON BEAM IS ASSUMED NORMALLY
INCIDENT ON THE LEFT SURFACE OF THE SLAB. WHEN AN ELECTRON
CROSSES A BOUNDARY(IMAGINARY) KNOWN AS A "SPLITTING PLANE" IT
SPLITS INTO N IDENTICAL PARTICLES EACH WITH STATISTICAL WEIGHT
GIVEN BY 1/N OF THE ORIGINAL VALUE. THE MONTE CARLO PROCEDURE IS
THEN CARRIED OUT INDEPENDENTLY ON EACH OF THESE FRACTIONAL PARTI-
CLES. A LIST OF THE QUANTITIES CALCULATED BY THIS PROGRAM IS GIVEN
IN VOL. 1, PAGES 259-260. EXCEPT FOR THE HISTORY SPLITTING
FEATURE, THIS PROGRAM IS STRUCTURED IN A MANNER SIMILAR TO THAT
OF "BIGSLAB".

COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,EGY,CPHIO,
15PHIO,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLM,
2XD,YD,ZD,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
421,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
DIMENSION TJMR(11)
CALL SETRUN
NOTDONE=0
PREFLOAT(NMAX)
COLLNO=0.
MTM=1
CALL SECOND(TIME)
TMR(1)=TIME
ICROSS(1)=ICROSS(2)=0
5 N=N+1
IF(ICROSS(1).EQ.0.AND.ICROSS(2).EQ.0)CALL SETHIS
IF(ICROSS(1).EQ.1.AND.ICROSS(2).EQ.0)CALL SPLIT(1)
IF(ICROSS(2).EQ.1)CALL SPLIT(2)
IF(IFLAG.EQ.1)GO TO 5
20 CALL PENET
CALL ENERGY
CALL SCORE
IF(IFLAG.GT.0) GO TO 6
IF(NCOUNT(1).GT.0)GO TO 400
IF(Z.GE.PCROSS(1).AND.ICROSS(1).EQ.0)CALL SPLIT(1)
400 IF(NCOUNT(2).GT.0)GO TO 401
IF(Z1.GT.PCROSS(2).AND.Z.LT.PCROSS(2).AND.ICROSS(2).EQ.0)
1CALL SPLIT(2)
401 CONTINUE
IF(IFLAG.GT.0) GO TO 6
IF(NCOUNT(1).EQ.0.AND.ICROSS(1).EQ.1)GO TO 402
IF(NCOUNT(2).EQ.0.AND.ICROSS(2).EQ.1)GO TO 402
402 IF(IFLAG.LT.0)GO TO 5
CALL ANGLES
GO TO 20
6 COLLNO=COLLNO+FLOAT(NCOLL)/FN
IF(NBATCH.EQ.0)GO TO 60
IF(ICROSS(1).EQ.0)GO TO 47
IF(ICROSS(2).EQ.0.AND.NCOUNT(1).EQ.10)GO TO 47
IF(ICROSS(1).EQ.1.AND.NCOUNT(1).EQ.10.AND.NCOUNT(2).EQ.10)GO TO 47
NOTDONE=1

```



```

G      7 60
47 C    INUE
      NOTDONE=0
      NTEST=NTRUE-NBATCH*(NTRUE/NBATCH)
      IF (NTEST.NE.0) GO TO 60
      MTM=MTM+1
      CALL SECOND(TIME)
      TIMR(MTM)=TIME
      DELT=TIME-TIMR(1)
      AVGT=DELT/FLOAT(MTM-1)
      TREMAIN=TLIM-DELT
      TLEFT=1.5*AVGT
      IPCT=10*(MTM-1)
      PRINT 75,IPCT,TREMAIN
75 FORMAT(1X,'THE PROBLEM IS',14,' PERCENT COMPLETE.',1X,
1,TIME TO FINISH IS',F12.3,'SECONDS')
      IF (IPCT.EQ.100) GO TO 100
      IF (TREMAIN.GE.TLEFT) GO TO 5
      GO TO 100
60 IF (NTRUE-NMAX) IS.55.55
55 IF (NOTDONE.E2.1) GO TO 5
2 FORMAT(1X,7E16.9)
100 PRINT 64,COLLND
      IF (LCMR.GT.0.AND.LCMR.LE.201) WRITE(2)((STO(K,L),K=1,7),L=1,201)
      IF (LCMR.GT.0.AND.LCMR.LE.201.AND.NPRINT.EQ.2) PRINT 2,
1((STO(K,L),K=1,7),L=1,LCMR)
      IF (LCMR.EQ.0) WRITE(2)((STO(K,L),K=1,7),L=1,201)
      CALL RANGET(XIRC)
64 FORMAT(//1X,'AVERAGE NUMBER OF COLLISIONS PER HISTORY = ',F7.3)
      IF (NBATCH.EQ.0) GO TO 200
      MTM=MTM-1
      PRINT 65,MTM,NBATCH
65 FORMAT(//1X,'THERE ARE ',14,' BATCHES OF',15,' HISTORIES')
      AVGT=0
      SUM=TIMR(1)
      DO 66 M=2,MTM
      TIMR(M)=TIMR(M)-SUM
      SUM=SUM+TIMR(M)
      AVGT=AVGT+TIMR(M)
      MM=M-1
66 PRINT 67,MM,TIMR(M)
87 FORMAT(1X,'BATCH NO.',14,' TIME = ',F6.3,' SECONDS')
      AVGT=AVGT/FLOAT(MTM1)
      PRINT 68,AVGT
88 FORMAT(//1X,'AVERAGE TIME PER BATCH = ',F6.3,' SECONDS')
200 PRINT 190
190 FORMAT(1,6X,'INRAN=',11X,'IRA=',12X,'IRC=')
      PRINT 195,INRAN,XIRA,XIRC
195 FORMAT(3(1X,O16))
      CALL SECOND(T1)
      CALL PROC
      CALL SECOND(T2)
      T3=T2-T1
      PRINT 196,T3
196 FORMAT(1X,'TIME REQUIRED FOR PROCESSING HISTORIES = ',F10.3,'SEC')
      STOP
      END
      SUBROUTINE SETRUN
      COMMON/MC/STO(17,201),NCOLL,CTHQ,CTH,STM,PHI,CPhi,SPHI,EGY,CPHIO,
1SPHIO,EO,SPAC,AMBDA,N,IFLAG,MAXCOLL,LCMR,STAB(200),ETAB(200),TLIM,
2HO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),AMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
      DIMENSION INCODE(3)
      DATA CPHIO,SPHIO,PI,ZMIN,ZMAX,N/1.0,0.0,3.14159265,0.0,1000.0,0/
      DATA ECUT,SPAC,XO,YO,ZO/O..800..3*0./

```

```

DATA THICK/11-3.
C 1.NMAX,MAXCOLL,NPRINT,INPT,NSLABS
C 2.TOTAL NO. OF MONTE CARLO HISTORIES
C 3.STORED ON TAPE OR DISK UNTIL PROCESSING TIME
C 4.MAXCOLL - MAXIMUM NUMBER OF COLLISIONS PER HISTORY TO BE ALLOWED
C 5.NPRINT - DEBUG PRINT CONTROL PARAMETER
C 6.NPRINT EQUAL TO ZERO SUPPRESSES DEBUG PRINTOUT
C 7.NPRINT NOT EQUAL TO ZERO ACTIVATES DEBUG PRINTOUT

READ 3,TLIM,CTHO,EO,ECUT
IF(NSLABS.EQ.0)GO TO 50
READ 3,(THICK(I),I=1,7)
DO 51 I=1,7
51 THICK(I)=THICK(I)*1.E-5
PCROSS(1)=THICK(2)
4 PCROSS(2)=THICK(6)
IF(INPT.GT.0)GO TO 40
INRAN=0
GO TO 45
40 READ 1,(INCODE(I),I=1,3)
IF(INCODE(1).EQ.1)READ 8,INRAN
IF(INCODE(2).EQ.1)READ 3,SFAC
IF(INCODE(3).EQ.1)READ 3,XO,YO,ZO
EO - INITIAL PARTICLE ENERGY
XO,YO,ZO - INITIAL PARTICLE COORDINATES
SFAC - STOPPING POWER FUDGE FACTOR
EFAC - ETA FUDGE FACTOR
45 PRINT 4,NMAX,MAXCOLL
PRINT 9,EO,ECUT,XO,YO,ZO,CTHO,CPHIO,SPHIO
PRINT 11,SPAC
PRINT 33,(THICK(I),I=1,11)
33 FORMAT(1X,'SLAB BOUNDARIES'/(1X,10E12.5))
11 FORMAT(1X,'STOPPING POWER FUDGE FACTOR (SFAC) =',F10.5)
9 FORMAT(1X,'EO (INITIAL ENERGY MEV)',E12.5//1X,'*EO (LOW ENERGY
1CUT OFF MEV) ',E12.5//1X,'*XO, YO, ZO. (SOURCE POINT COORDINATES)
3 ',E12.5//1X,'*CTHO (INCIDENT POLAR COSINE) ',E12.5//1X,'*CPHIO
4,SPHIO (INCIDENT AZIMUTH) ',E12.5)
8 FORMAT(018)
1 FORMAT(610)
3 FORMAT(8F10.0)
4 FORMAT(11//1X,'NMAX (NO. OF HISTORIES) ',I6//1X,'MAXCOLL (MAXIMUM
1 ALLOWED NUMBER OF COLLISIONS) ',I3)
10 N=0
NTRUE=0
LCHR=0
XIRA=XIRC=INRAN
NBATCH=NMAX/10
CALL RANSET(XIRA)
CALL ESET
REWIND 2
DO 20 K=1,7
DO 20 L=1,201
20 STO(K,L)=0.0
RETURN
END
SUBROUTINE SETHIS
COMMON/MC/STD(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,EO,SFAC,AMEDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
NCOUNT(1)=NCOUNT(2)=-1
WF=1.
ICROSS(1)=ICROSS(2)=0
NCOLL=0
PATH=0.
S=0.

```

```

X=Y
Y=
Z=ZO
CPHI=CPHIO
SPHI=SPHIO
PHI=0.0
CTH=CTHO
STH=STHIO(1.-CTH**2)
EGV=EO
IE=1
NTRUE=NTRUE+1
CALL CROSS
IFLAG=0
CALL SCORE
RETURN
END
SUBROUTINE PENET
COMMON/MC/STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGV,CPHIO,
1SPHIO,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
X1=X
Y1=Y
Z1=Z
4 RB=RAMF(Q)
IF(RB.GT.0.)GO TO 5
GO TO 4
5 S=-ALOG(RB)*AMBDA
Z=Z1+S*CTH
IF(Z.GT.ZMIN) GO TO 10
Z=ZMIN
S=(ZMIN-Z1)/CTH
IFLAG=1
10 X=X1+S*STH*CPHI
Y=Y1+S*STH*SPHI
PATH=PATH+S
NCOLL=NCOLL+1
IF(NCOLL.EQ.MAXCOLL) IFLAG=1
IF(Z.GT.ZMAX)IFLAG=1
RETURN
END
SUBROUTINE ESET
COMMON/MC/STOI(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGV,CPHIO,
1SPHIO,EO,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRINT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
DATA PI,XL2/3.14159265,693147181/
DATA ZA,AA,AVOG,RO,DNSTY,EACIT,FCC/13.,26.98,6.025E23,2.8179E-13,
12.7,163.E-6,.15360628/
ENTAB(1)=1.0
DE=.01
DO 1 II=2,91
1 ENTAB(II)=ENTAB(II-1)*DE
DE=.001
DO 2 II=92,181
2 ENTAB(II)=ENTAB(II-1)*DE
DO 3 III=1,161
3 E=ENTAB(III)/.511
E1=(1.+E)*(1.+E)
E2=E*(2.+E)
BETA=SQRT(E2/E1)
XLG=ALOG(E*E2/12.*(EXCIT/.511)**2)
FM=1.-BETA*BETA*(.125+E*(2.+E+1.)*XL2)/E1
STAB(1)=(FCC*ZA/AA/BETA**2)*(XLG+FM)*SFAC

```

```

ETAB(11)=0.5*(ZA**((1./3.)/(121.245)**2/E2*(1.13+3.76*(ZA/137.))**2.
1E1))
SI AB(11)=ZA*(1.+ZA)*AVOG*RO*RO*ZA/AA*E1/(E2+E2)/(ETAB(11),
1*(1.+5*ETAB(11)))
ETAB(11)=.035276
3 IF(11.GT.91)ETAB(11)=.103378
IF(NPRT.EQ.0)RETURN
PRINT 75
75 FORMAT(1X,'CROSS SECTION AND STOPPING POWER TABLES',//,1X,'ENERGY(M
1EV) STOPPING POWER ETA SIGMA*//)
DO 76 11=1,181
76 PRINT 77,ETAB(11),STAB(11),ETAB(11),SIGTAB(11)
77 FORMAT(1X,F10.6,3E16.9)
RETURN
END
SUBROUTINE ANGLES
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STM,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,ED,SPAC,AMSDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
DATA TMOPI/6.283185307/
RD=2.*RANF(0)
COM=RD*(1.+ETA)-ETA)/(RD+ETA)
SOM=SQRT(1.-COM**2)
RC=RANF(0)
RHO=TMOPI*RC
CRHO=COS(RHO)
SRHO=SIN(RHO)
STH=STM
CTH=CTH
CTH=CTH*COM+STM*STM+SOM*CRHO
STM=SQRT(1.-CTH**2)
IF(STM.EQ.0.0.OR.STM.EQ.0.0) GO TO 1
CI=(COM-CTH*CTH)/(STM*STM)
SI=SRHO*SOM/STM
CPHI=CPHI
SPHI=SPHI
CPHI=CPHI*CI-SPHI*SI
SPHI=CPHI*SI+SPHI*CI
GO TO 2
1 CPHI=CRHO
SPHI=SRHO
2 CONTINUE
PHI=ATAN2(SPHI,CPHI)
IF(PHI.LT.0.0)PHI=PHI+TMOPI
RETURN
END
SUBROUTINE SCORE
COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STM,PHI,CPHI,SPHI,EGY,CPHIO,
1SPHIO,ED,SPAC,AMSDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
DO 100 1=3,5
IF(NCOLL.EQ.0)GO TO 70
96 IF(CTH)97,97,99
99 IF(21.LT.THICK(1).AND.Z.GT.THICK(1))GO TO 70
GO TO 100
97 IF(2.LE.THICK(1).AND.Z1.GT.THICK(1))GO TO 70
GO TO 100
70 LCHR=LCHR+1
STO(1,LCHR)=FLOAT(1)
IF(NCOLL.EQ.0)STO(1,LCHR)=100.
STO(2,LCHR)=CTHO
STO(3,LCHR)=CTH

```

```

      STO(4,LCHR)=STM
      5( 3,LCHR)=PHI
      ST(6,LCHR)=EGY
      STO(7,LCHR)=WF
      IF(LCHR.LT.201)GO TO 95
      WRITE(2)((STO(MM,LL),MM=1,7),LL=1,LCHR)
      IF(INPRT.EQ.2)PRINT 2,((STO(MM,LL),MM=1,7),LL=1,LCHR)
      2 FORMAT(1X,7E16.9)
      IF(LCHR.EQ.0)WRITE(2)((STO(K,L),K=1,7),L=1,201)
      DO 1 MM=1,7
      DO 1 LL=1,LCHR
      1 STO(MM,LL)=0.
      LCHR=0
      95 IF(NCOLL.EQ.0)RETURN
      100 CONTINUE
      RETURN
      END
      SUBROUTINE LEP(Y,X,N)
      DIMENSION Y(1)
      Y(1)=1.
      IF(N)1,1,2
      1 RETURN
      2 Y(2)=X
      IF(N-1)1,1,3
      3 DO 4 I=2,N
      G=X*Y(I)
      4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)
      RETURN
      END
      SUBROUTINE STATS(XVAL,ISIG,N,K,IB)
      DIMENSION XVAL(1),ISIG(1),AVE(10),SIG(10)
      DIMENSION ECAVE(4200),ECSIG(4200)
      IF(18.NE.1)GO TO 2
      DO 1 I=1,N
      AVE(I)=XVAL(I)
      ISIG(I)=99
      1 SIG(I)=0.0
      10 DO 100 I=1,N
      INK=I-1
      ECAVE(INK)=AVE(I)
      100 ECSIG(INK)=SIG(I)
      K=K+N
      RETURN
      2 XNSORS=18
      DO 200 I=1,N
      INK=I-1
      AVE(I)=ECAVE(INK)
      SIG(I)=ECSIG(INK)
      200 SIG(I)=ECSIG(INK)
      DO 3 I=1,N
      SIG(I)=(XNSORS-2.0)/(XNSORS-1.0)*SIG(I)+(XVAL(I)-AVE(I))*2/XNSORS
      AVE(I)=((XNSORS-1.0)*AVE(I)+XVAL(I))/XNSORS
      XVAL(I)=AVE(I)
      ERROR=SQRT(SIG(I)/XNSORS)
      ISIG(I)=99
      IF(AVE(I).EQ.0.0)GO TO 3
      ISIG(I)=100.0*ERROR/ABS(AVE(I))+.50001
      IF(1SIG(I).GT.99)ISIG(I)=99
      3 CONTINUE
      GO TO 10
      END
      SUBROUTINE ENERGY
      COMMON/MC/STO(7,201),NCOLL,CTHO,CTH,STH,PHI,CPHI,SPHI,EGY,CPHIO,
      1SPHIQ,EO,SFAC,AMBOA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
      2XD,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),MMAX,ZMIN,ZMAX,IE,INRAN,XIPA,
      3,NPRT,TLAST,CTMC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
      4Z1,WF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE

```

```

1 IF EGY=1)1)1.2.2
2 IF >FIX(100.*(1.-EGY))
3 GO 0.3
1 IF EGY.LT.0.01)EGY=.01
1E-91+FIX(1000.*(1.-EGY))
3 DE=STAB(IE)*S
EGY=EGY-DE
IF EGY.GT.ECUT)GO TO 4
IF LAG=1
ENTRY CROSS
EGY=ECUT
ETA=ETAB(IE)
ANBDA=1./SIGTAB(IE)
RETURN
END
SUBROUTINE ASSOC(PL,CX, SX,P)
DIMENSION PL(10),P(9,9)
IF(SX.GT.0.0)GO TO 1
DO 2 L=1,9
DO 2 M=1,9
2 P(L,M)=0.0
RETURN
1 P(1,1)=SX
P(2,1)=3.*CX*SX
P(2,2)=2.*CX/SX*P(2,1)-6.*PL(3)
M=1
LOW=M+1
DO 10 L=LOW,8
10 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L*M)*P(L-1,M))/FLOAT(L-M+1)
P(3,2)=2.*CX/SX*P(3,1)-12.*PL(4)
DO 100 M=2,8
LOW=M+1
IF(LOW.EQ.9)GO TO 51
DO 50 L=LOW,8
50 P(L+1,M)=((2*L+1)*CX*P(L,M)-(L*M)*P(L-1,M))/FLOAT(L-M+1)
51 P(LOW,M+1)=((2*M)*P(LOW,M)+CX/SX-(LOW+M)*P(LOW,M-1))/
IF(M.EQ.8)GO TO 100
P(LOW+1,M+1)=(2*M)*P(LOW+1,M)+CX/SX-((LOW+M+1)*P(LOW,M+2))+
1P(LOW+1,M-1)
100 CONTINUE
RETURN
END
SUBROUTINE SPLIT(K)
END
C "SPLIT" PERFORMS THE HISTORY SPLITTING. IT IS CALLED WHEN A
C SPLITTING PLANE IS CROSSED.
C
COMMON/NC/STO(17,201),NCOLL,CTHO,CTH,STH,PHI,CPI,SPHI,EGY,CPIIO,
1SPHI0,EQ,SFAC,AMBA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2X0,Y0,Z0,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA,
3,NPRNT,TLAST,CTHC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,MF,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
DIMENSION XSV(2),YSV(2),ZSV(2),ESV(2),CTS(2),STS(2),CPS(2),SPS(2),
1PSV(2),IESV(2),NCSV(2),WFO(2)
DATA WFO/1,...,1/
IF(NCOUNT(K).LT.10)GO TO 1
IF(K.EQ.1)NCOUNT(1)=NCOUNT(2)+1
NCOUNT(K)=1
IF LAG=1
ICROSS(K)=0
RETURN
1 IF(NCOUNT(K).GE.0)GO TO 10
XSV(K)=X
YSV(K)=Y
ZSV(K)=Z
ESV(K)=EGY

```

```

CTH(K)=CTH
S(4)=STH
CP(K)=CPHI
SPS(K)=SPHI
PSV(K)=PATH
IESV(K)=IE
NCSV(K)=NCOLL
IFLAG=-1
NCOUNT(K)=NCOUNT(K)+1
ICROSS(K)=1
RETURN
10 X=XSV(K)
Y=YSV(K)
Z=ZSV(K)
EGY=ESV(K)
CTH=CTS(K)
STH=STS(K)
CPHI=CPS(K)
SPHI=SPS(K)
PATH=PSV(K)
WF=WFO(K)/10.
N=N-1
IE=IESV(K)
S=0.
CALL CROSS
IFLAG=0
NCOLL=NCSV(K)
NCOUNT(K)=NCOUNT(K)+1
RETURN
END
SUBROUTINE PROC
COMMON/MC/STD(7,201),NCOLL,CTHD,CTH,STH,PHI,CPHI,SPHI,EGY,CPHI0,
15PHI0,ED,SFAC,AMBDA,N,IFLAG,MAXCOLL,LCHR,STAB(200),ETAB(200),TLIM,
2XO,YO,ZO,X,Y,Z,ECUT,CLAB,THICK(11),NMAX,ZMIN,ZMAX,IE,INRAN,XIRA
3,NPRINT,TLAST,CTMC,NBATCH,ETA,PATH,S,SIGTAB(200),ENTAB(200),XIRC,
4Z1,W,NCOUNT(2),ICROSS(2),PCROSS(2),NTRUE
5DIMENSION TRANC(10,10),TRANF(10,10),BACKC(10,10),BACKF(10,10),
6APLUS(9,9,10),AMINUS(9,9,10),PLUS(9,9,10),MINUS(9,9,10),
7ALLEG(10,10),LEG(10,10),ENERGY(10),YCOF(9,9),ANGLE(10),PL(10),
8POLM(9,9),CP(9),SP(9),SUM(10),TEMP(10),ISIG(10),ITP(9,9,10),
9ATM(9,9,10),ISGEX(10,10),IPR(10)
DO 1 L=1,9
FL=(2.*FLOAT(L)+1.)/6.283185307
DO 2 M=1,L
LM=L-M+1
LM=L+M
FI=1.
DO 3 J=LMI,LM
FI=FI*FLOAT(J)
2 YCOF(L,M)=SORT(FL/FI)
1 CONTINUE
REWIND 2
NIN=NMAX/10
FN=FLOAT(MIN)
DE=ED/10.
ENERGY(1)=EO-DE
ANGLE(1)=.1
DO 4 K=2,10
ANGLE(K)=ANGLE(K-1)+.1
4 ENERGY(K)=ENERGY(K-1)-DE
1PLANE=2
100 REWIND 2
IB=NH-NHIST-0
1PLANE=1PLANE+1
PRINT 777,THICK(1PLANE)
777 FORMAT('10.40X,DETECTOR PLANE AT=.E12.5)

```

```

DO 40 I=1,10
  IW=0
  DO 40 J=1,10
    TRANC(I,J)=TRANF(I,J)+BACKC(I,J)+BACKF(I,J)*O.
  41 ALEG(I,J)=TLEG(I,J)*O.
  DO 40 J=1,9
    DO 40 K=1,9
      40 APLUS(J,K,I)=AMINUS(J,K,I)+PLUS(J,K,I)+TMINUS(J,K,I)*O.
5000 READ(2)((STO(K,L),K=1,7),L=1,201)
DO 90 L=1,201
  IF(STO(1,L).EQ.O.)GO TO 11
  IF(STO(1,L).LT.10.)GO TO 10
  NH=NH+1
  11 NHIST=NHIST+1
  GO TO 50
  10 IPL=FIX(STO(1,L)+1.E-6)
  IF(IPL.NE.IPLANE)GO TO 90
  W=STO(7,L)
  CLASSIFY EMERGENT ANGLE
  CA=ABS(STO(3,L))
  DO 80 J=1,10
    80 IF(CA.LT.ANGLE(J))GO TO 81
    J=10
  81 JC=J
  CLASSIFY EMERGENT ENERGY
  DO 82 K=1,10
    82 IF(STO(6,L).GT.ENERGY(K))GO TO 83
    K=10
  83 KC=K
  IPR(KC)=1
  CTN POSITIVE OR NEGATIVE?
  CTMET=STO(3,L)
  STMET=STO(4,L)
  PHI=STO(5,L)
  CALL LEP(PL,CTMET,9)
  DO 70 M=1,9
    CP(M)=COS(M*PHI)
  70 SP(M)=SIN(M*PHI)
  CALL ASSOC(PL,CTMET,STMET,PLM)
  IF(CTMET)84,85,85
  84 BACKC(JC,KC)=BACKC(JC,KC)+W/FN
  BACKF(JC,KC)=BACKF(JC,KC)-W/CTMET/FN
  DO 86 LL=1,10
    86 ALEG(LL,KC)=ALEG(LL,KC)+PL(LL)*W/FN
    DO 87 LL=1,9
      DO 88 M=1,LL
        FM=YCOF(LL,M)*PLM(LL,M)/FN
        APLUS(LL,M,KC)=APLUS(LL,M,KC)+FM*CP(M)*W
  88 AMINUS(LL,M,KC)=AMINUS(LL,M,KC)+FM*SP(M)*W
  87 CONTINUE
  GO TO 90
  85 TRANC(JC,KC)=TRANC(JC,KC)+W/FN
  TRANF(JC,KC)=TRANF(JC,KC)+W/FN/CTMET
  DO 89 LL=1,10
    89 TLEG(LL,KC)=TLEG(LL,KC)+PL(LL)*W/FN
    DO 99 LL=1,9
      DO 91 M=1,LL
        FM=YCOF(LL,M)*PLM(LL,M)/FN
        TPLUS(LL,M,KC)=TPLUS(LL,M,KC)+FM*CP(M)*W
  91 TMINUS(LL,M,KC)=TMINUS(LL,M,KC)+FM*SP(M)*W
  99 CONTINUE
  GO TO 90
  50 IF(NHIST.LE.NNH)GO TO 90
  NHIST=1
  KPUT=1
  18=18+1

```



```

DO 150 MM=1,4
  DO 14 I=1,10
    IF (IB.LT.10) GO TO 154
    PRINT 220
    GO TO (114,124,134,144),MM
  114 PRINT 230
    GO TO 153
  124 PRINT 231
    GO TO 153
  134 PRINT 232
    GO TO 153
  144 PRINT 233
  153 PRINT 234,(ANGLE(I),I=1,10)
  230 FORMAT(///45X,'POSITIVE CURRENT-//')
  231 FORMAT(///45X,'NEGATIVE CURRENT-//')
  232 FORMAT(///45X,'POSITIVE FLUX-//')
  233 FORMAT(///45X,'NEGATIVE FLUX-//')
  234 FORMAT(45X,'EMERGENT POLAR COSINE-/9X,10E12.5/3X,'E(MEV)-')
  154 DO 164 K=1,10
    DO 165 I=1,10
      GO TO (115,125,135,145),MM
    115 TEMP(I)=TRANC(I,K)
      TRANC(I,K)=0.
      GO TO 105
    125 TEMP(I)=BACKC(I,K)
      BACKC(I,K)=0.
      GO TO 105
    135 TEMP(I)=TRANF(I,K)
      TRANF(I,K)=0.
      GO TO 105
    145 TEMP(I)=BACKF(I,K)
      BACKF(I,K)=0.
    165 CONTINUE
    CALL STATS(TEMP,ISIG,10,KPUT,18)
    IF (IB.LT.10) GO TO 104
    DO 156 I=1,10
      SUM(I)=SUM(I)+TEMP(I)
      IF (IPR(K).EQ.0) GO TO 104
      PRINT 221,ENERGY(K),(TEMP(I),ISIG(I),I=1,10)
    166 CONTINUE
    104 CONTINUE
    IF (IB.EQ.10) PRINT 235,(SUM(I),I=1,10)
    235 FORMAT(//2X,'TOTALS ',10(E12.5))
    150 CONTINUE
    DO 1700 I=1,9
      DO 1700 J=1,9
        DO 1701 K=1,10
          TEMP(K)=TPLUS(I,J,K)
          TPLUS(I,J,K)=0.
          CALL STATS(TEMP,ISIG,10,KPUT,18)
          IF (IB.LT.10) GO TO 1700
          DO 1702 K=1,10
            TPLUS(I,J,K)=TEMP(K)
          1702 ITP(I,J,K)=ISIG(K)
          1700 CONTINUE
          DO 1710 I=1,9
            DO 1710 J=1,9
              DO 1711 K=1,10
                TEMP(K)=TMINUS(I,J,K)
                TMINUS(I,J,K)=0.
                CALL STATS(TEMP,ISIG,10,KPUT,18)
                IF (IB.LT.10) GO TO 1710
                DO 1712 K=1,10
                  TMINUS(I,J,K)=TEMP(K)
                1712 ITM(I,J,K)=ISIG(K)
                1710 CONTINUE

```

```

DO 720 I=1,10
A
  FAL=SQRT((2.*AL-1.)/12.56637061)
  DO 1721 J=1,10
    TEMP(J)=TLEG(I,J)
1721 TLEG(I,J)=0.
    CALL STATS(TEMP,ISIG,10,KPUT,18)
    IF(18.LT.10)GO TO 1720
    IF(MA.NE.NMAX)STOP 7
    DO 1722 J=1,10
      TLEG(I,J)=FAL*TEMP(J)
1722 ISGEX(I,J)=ISIG(J)
1720 CONTINUE
    IF(18.EQ.10)PRINT 499,NH
    IF(18.LT.10)GO TO 1730
220 FORMAT(///)
221 FORMAT(1X,F7.3,1X,10(F9.5,13))
499 FORMAT(1X,*TOTAL NUMBER OF HISTORIES PROCESSED = *,I10)
PRINT 1523
1523 FORMAT(///45X,*LEGENDRE COEFFICIENTS OF TRANSMISSION*/
14X,*E(MEV) L= 4X,*0*,11X,*1*,11X,*2*,11X,*3*,11X,*4*,
21X,*5*,11X,*6*,11X,*7*,11X,*8*,11X,*9*)
DO 1626 K=1,10
  IF(IPR(K).EQ.0)GO TO 1626
  PRINT 526,ENERGY(K),(TLEG(LL,K),LL=1,10)
1626 CONTINUE
PRINT 651
DO 1525 K=1,10
  IF(IPR(K).EQ.0)GO TO 1525
  PRINT 524,ENERGY(K),(ISGEX(LL,K),LL=1,10)
1525 CONTINUE
DO 1345 KE=1,10
  IF(1PR(KE).EQ.0)GO TO 1345
  PRINT 1347,ENERGY(KE)
1347 FORMAT(///1X,*ENERGY = *,F10.5,* MEV*/1X,*SPHERICAL HARMONICS COEFF
ICIENTS FOR TRANSMISSION, TPLUS*/
DO 1343 LL=1,9
1343 PRINT 349,LL,(TPLUS(LL,M,KE),M=1,LL)
PRINT 651
DO 1652 LL=1,9
1652 PRINT 653,LL,(ITP(LL,M,KE),M=1,LL)
1346 FORMAT(1X,*SPHERICAL HARMONICS COEFFICIENTS FOR TRANSMISSION, TMIN
1US*/
DO 1342 LL=1,9
1342 PRINT 349,LL,(TMINUS(LL,M,KE),M=1,LL)
PRINT 651
DO 1654 LL=1,9
1654 PRINT 653,LL,(ITM(LL,M,KE),M=1,LL)
1345 CONTINUE
1730 DO 700 I=1,9
  DO 700 J=1,9
    DO 701 K=1,10
      TEMP(K)=APLUS(I,J,K)
701 APLUS(I,J,K)=0.
      CALL STATS(TEMP,ISIG,10,KPUT,18)
      IF(18.LT.10)GO TO 700
      DO 702 K=1,10
        APLUS(I,J,K)=TEMP(K)
702 ITP(I,J,K)=TEMP(K)
700 CONTINUE
      DO 710 I=1,9
        DO 710 J=1,9
          DO 711 K=1,10
            TEMP(K)=AMINUS(I,J,K)
711 AMINUS(I,J,K)=0.

```

```

C 11  STATES(TEMP,ISIG,10,KPUT,18)
  IF (B.LT.10)GO TO 710
  DO 12 K=1,10
    AMINUS(I,J,K)=TEMP(K)
  712  ITM(I,J,K)=ISIG(K)
  710  CONTINUE
  DO 720 I=1,10
    DO 721 J=1,10
      TEMP(J)=ALEG(I,J)
  721  ALEG(I,J)=0.
  CALL STATES(TEMP,ISIG,10,KPUT,18)
  IF (B.LT.10)GO TO 720
  DO 722 J=1,10
    ALEG(I,J)=TEMP(J)
  722  ISGEX(I,J)=ISIG(J)
  720  CONTINUE
  IF (B.LT.10)GO TO 90
  PRINT LEGENDRE COEFFICIENTS
  523  FORMAT(/45X,'LEGENDRE COEFFICIENTS OF BACKSCATTER=4X,E(MEV) L=
    11X,80,11X,10,11X,20,11X,30,11X,40,11X,50,11X,60,11X,70,
    211X,80,11X,90')
  DO 625 LL=1,10
    AL=LL
    FAL=SQRT((2.*AL-1.)/12.56837061)
    DO 626 K=1,10
      ALEG(LL,K)=ALEG(LL,K)*FAL
  625  CONTINUE
  PRINT 523
  DO 626 K=1,10
    IF (IPR(K).EQ.0)GO TO 628
    PRINT 526,ENERGY(K),(ALEG(LL,K),LL=1,10)
  626  CONTINUE
  526  FORMAT(3X,F7.3,1X,10E12.5)
  PRINT 651
  DO 525 K=1,10
    IF (IPR(K).EQ.0)GO TO 525
    PRINT 524,ENERGY(K),(ISGEX(LL,K),LL=1,10)
  525  CONTINUE
  524  FORMAT(3X,F7.3,1X,10I4)
  DO 345 KE=1,10
    IF (IPR(KE).EQ.0)GO TO 345
  3461  PRINT 347,ENERGY(KE)
  347  FORMAT(/1X,'ENERGY = ',F10.5,' MEV',1X,'SPHERICAL HARMONICS COEFF
    11CIENTS - APLUS=')
  DO 343 LL=1,9
    343  PRINT 349,LL,(APLUS(LL,M,KE),M=1,LL)
  349  FORMAT(1X,L=,15,1X,9E12.5)
  PRINT 651
  651  FORMAT(1X,'PERCENT ERRORS IN ABOVE ITEMS=')
  DO 652 LL=1,9
    652  PRINT 653,LL,(ITP(LL,M,KE),M=1,LL)
  653  PRINT 348
  348  FORMAT(1X,'SPHERICAL HARMONICS COEFFICIENTS - AMINUS=')
  DO 342 LL=1,9
    342  PRINT 349,LL,(AMINUS(LL,M,KE),M=1,LL)
  PRINT 651
  DO 654 LL=1,9
    654  PRINT 653,LL,(ITM(LL,M,KE),M=1,LL)
  653  CONTINUE
  653  FORMAT(1X,L=,15,1X,9I4)
  GO TO 200
  90  CONTINUE
  GO TO 5000
  200  IF (IPLANE.LT.5)GO TO 100
  RETURN
  END

```

Program Listing

- CHDP -

(Reference: Vol. I, Section IV.B.5)


```

      RE=0.46*(IF(I1,I=1,NX)
48 F( AT(4012)
C
C CONVERT PERCENT ERRORS TO ABSOLUTE ERROR ESTIMATES
C
      DO 47 I=1,NX
      DY(I,L)=.01*FLOAT(IF(I1))=ED(I,L)
47 IF(I1)=0
100 CONTINUE
      DO 101 I=1,100
101 F(I)=Y(I)=0.
45 FORMAT(16F5.0)
      DO 200 L=1,3
      TO=EO(L)
      IMAX=NP(L)
      DO 160 I=1,IMAX
      XP(I)=X(I,L)
160 YP(I)=ED(I,L)
      PRINT 72,L
      PRINT 71,(XP(I),YP(I),I=1,IMAX)
      DN=FLOAT(IMAX+1)
      SM=FLOAT(IMAX+1)-DN
      DO 149 I=1,IMAX
149 S(I)=DY(I,L)
      DO 150 K=1,3
C
C CALL LEAST SQUARES SPLINE SMOOTHING ROUTINE
C
      CALL ICSSCU(XP,YP,S,IMAX,SM,NV,C,1C,MK,IER)
      OX=AMAX(L)/50.
      XX=0.5*DX
      DO 40 N=1,50
      DO 41 I=2,IMAX
41 IF(XX.LT.XP(I))GO TO 42
42 D=XX-XP(I-1)
      I=I-1
      Y=((C(I,3)*D+C(I,2))*D+C(I,1))*D+NY(I)
      X(N,L)=XX
      F(N)=Y
      IF(K.GT.1)GO TO 130
      XT=XX*1.E-3
C
C CALL TABATA PROGRAM TO CALCULATE CURRENT VALUES
C
      CALL DEPOSI(CU,VY,TO,ZE,AM,MODE,XT)
      YTN)=VY
      CUR(N)=CU
130 XX=XX+DX
      IF(XX.GE.XP(IMAX))GO TO 48
40 CONTINUE
48 PRINT 73,SM
      PRINT 71,(X(N,L),FIN),N=1,51)
      IF(K.GT.1)GO TO 131
      PRINT 81
      PRINT 71,(X(N,L),YT(N),N=1,51)
      FORMAT(/1X,'TABATA ALGORITHM RESULTS'//)
      PRINT 82
      PRINT 71,(X(N,L),CUR(N),N=1,51)
      PRINT 71,(X(N,L),CUR(N),N=1,51)
82 FORMAT(1X,'TABATA CURRENT VALUES')
131 XLO=XP(1)
      XHI=XX-DX
C
C INTEGRATE SMOOTHED CURRENTS TO GET TOTAL CHARGE DEPOSITED
C
      CALL SIMP(F1,XLO,XHI,1.E-5,AREA,ERR,AR,IFL)
      PRINT 9,AREA

```

```

9 FORMAT(1X,'INTEGRAL=',E12.5)
150 S( N=DN
200 C( INUE
72 FORMAT(1X,'DATA SET NO.=12/1X,'RAW DATA=)
73 FORMAT(1X,'1X,'INTERPOLATED VALUES',5X,'SM=',F10.5)
71 FORMAT(5I2F10.3,6X))
37 FORMAT(16F5.0)
DO 300 L=1,6
NM=NM*(L)
300 READ 37,(ECUR(N,L),N=1,NM)
C
C EXPERIMENTAL CURRENT DEPOSITION DATA
C
DO 301 L=2,6
301 ECUR(1,L)=ECUR(1,L)
DO 302 L=4,6
302 ECUR(2,L)=ECUR(2,L)
XP(1)=10.7
XP(9)=95.6
XP(17)=175.
XP(25)=259.9
XP(33)=335.5
XP(41)=420.4
XP(49)=497.9
XP(57)=582.8
XP(64)=655.0
DO 303 N=2,8
XP(N)=XP(N-1)+10.7
XP(N+8)=XP(N+7)+10.7
XP(N+16)=XP(N+15)+10.7
XP(N+24)=XP(N+23)+10.7
XP(N+32)=XP(N+31)+10.7
XP(N+40)=XP(N+39)+10.7
XP(N+48)=XP(N+47)+10.7
XP(N+56)=XP(N+55)+10.7
XP(64)=655.
303 XP(N+63)=XP(N+62)+10.7
DO 350 L=1,6
NM=NM*(L)
DO 351 N=1,NM
XD(N)=XP(N)
YD(N)=ECUR(N,L)
351 ERD(N)=ERROR(N,L)
DN=SORT(2,'FLOAT(NM+1))
SN=FLOAT(NM+1)-DN
C
C SMOOTH THE EXPERIMENTAL DATA
C
DO 352 K=1,3
CALL ICSSCU(XD,YD,ERD,NM,SM,WY,C,IC,WK,IER)
PRINT 360,EGY(L),SM
PRINT 71,({XD(N),WY(N),N=1,NM)
360 FORMAT(1X,'SMOOTHED EXPERIMENTAL DATA, SOURCE ENERGY=',F5.3,'ME
1V=3X,'SMOOTHING PARAMETER=',F8.3)
352 SM=SM+DN
TD=EGY(L)
MODE=1
C
C OBTAIN CURRENT VALUES FROM TABATA PROGRAM
C
DO 362 N=1,NM
XX=XD(N)+1.E-3
CALL DEPOSI(CU,YY,TO,ZE,AW,MODE,XX)
MODE=2
XX=(XD(N)-10.7)+1.E-3
CALL DEPOSI(CV,YY,TO,ZE,AW,MODE,XX)

```

362 WY(N)=CV-CU

PR : 81

PR : 71,(XDIN),WY(N),N=1,NM)

350 CONTINUE

STOP

END

IMSL ROUTINE NAME - ICSSCU

COMPUTER - CDC/SINGLE

LATEST REVISION - JUNE 1, 1980

PURPOSE - CUBIC SPLINE DATA SMOOTHER

USAGE - CALL ICSSCU (X,F,DF,NX,SM,Y,C,IC,WK,IER)

ARGUMENTS X - VECTOR OF LENGTH NX CONTAINING THE ABSCISSAE OF THE NX DATA POINTS (X(1),F(1)) I=1,...., NX. (INPUT) X MUST BE ORDERED SO THAT X(1) .LT. X(I+1).

F - VECTOR OF LENGTH NX CONTAINING THE ORDINATES (OR FUNCTION VALUES) OF THE NX DATA POINTS. (INPUT)

DF - VECTOR OF LENGTH NX. (INPUT) DF(1) IS THE RELATIVE WEIGHT OF DATA POINT 1 (SEE PARAMETER SM BELOW).

NX - NUMBER OF ELEMENTS IN X, F, DF, AND Y. (INPUT) NX MUST BE GE. 2.

SM - A NON-NEGATIVE NUMBER WHICH CONTROLS THE EXTENT OF SMOOTHING. (INPUT) THE SPLINE FUNCTION S IS DETERMINED SUCH THAT THE SUM FROM 1 TO NX OF ((S(X(I))-F(1))/DF(1))**2 IS LESS THAN OR EQUAL TO SM, WHERE EQUALITY HOLDS UNLESS S DESCRIBES A STRAIGHT LINE.

Y,C - SPLINE COEFFICIENTS. (OUTPUT) Y IS A VECTOR OF LENGTH NX. C IS AN NX-1 BY 3 MATRIX. THE VALUE OF THE SPLINE APPROXIMATION AT T IS S(T) = ((C(1,3)+D+C(1,2))*D+C(1,1))*D+Y(1) WHERE X(1) .LE. T .LT. X(I+1) AND D = T-X(1).

IC - ROW DIMENSION OF MATRIX C EXACTLY AS SPECIFIED IN THE DIMENSION STATEMENT IN THE CALLING PROGRAM. (INPUT)

WK - WORK AREA VECTOR OF LENGTH 7*NX+14.

IER - ERROR PARAMETER. (OUTPUT) TERMINAL ERROR IER = 129, IC IS LESS THAN NX-1 IER = 130, NX IS LESS THAN 2 IER = 131, INPUT ABSCISSAE ARE NOT ORDERED SO THAT X(1) .LT. X(2) LT. X(NX)

PRECISION/HARDWARE - SINGLE AND DOUBLE/H32 SINGLE/H36,H48,H60

REQD. IMSL ROUTINES - UERTST,UGETIO

NOTATION - INFORMATION ON SPECIAL NOTATION AND CONVENTIONS IS AVAILABLE IN THE MANUAL INTRODUCTION OR THROUGH IMSL ROUTINE UHELP

REMARKS 1. THE ROUTINE PRODUCES A NATURAL CUBIC SPLINE. HENCE,


```

C      THE SECOND DERIVATIVE OF THE SPLINE FUNCTION S AT
C      X(1) AND X(NX) IS ZERO.
C      2. FOR EACH SET OF DATA POINTS THERE EXISTS A MAXI
C      VALUE FOR THE SMOOTHING PARAMETER. LET US CALL THIS
C      MAXIMUM VALUE SM. IT IS DEFINED BY THE FOLLOWING
C      FORMULA:
C      SM = THE SUM FROM I EQUAL 1 TO NX OF
C      ((Y(I)-F(I))/DF(I))
C      WHERE Y IS THE SET OF FUNCTION VALUES DEFINING THE
C      STRAIGHT LINE WHICH BEST APPROXIMATES THE DATA IN
C      THE LEAST SQUARES SENSE (WITH WEIGHTS DF).
C      COPYRIGHT - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C      WARRANTY - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C      APPLIED TO THIS CODE. NO OTHER WARRANTY,
C      EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C      SUBROUTINE ICSSCU (X,F,DF,NX,SM,Y,C,IC,WK,IER)
C      INTEGER
C      REAL
C      X(1),F(1),DF(1),Y(1),C(IC,1),WK(7,1)
C      I,J,M2,NP1,NP3
C      E,FF,F2,G,H,HMG,ONE,ONEDH,ONED3,P,SM,
C      TWOD3,ZERO
C      TWOD3=.666666666666667/
C      ONED3=.333333333333333/
C      ZERO/0.0,ONE/1.0/
C      FIRST EXECUTABLE STATEMENT
C      IER = 0
C      CHECK ERROR CONDITIONS
C      IF (IC .LT. NX-1) GO TO 65
C      IF (NX .LT. 2) GO TO 70
C      M2 = NX+2
C      NP1 = NX+1
C      WK(1,1) = ZERO
C      WK(1,2) = ZERO
C      WK(2,NP1) = ZERO
C      WK(3,M2) = ZERO
C      WK(3,NP1) = ZERO
C      WK(6,1) = ZERO
C      WK(6,2) = ZERO
C      WK(6,M2) = ZERO
C      WK(6,NP1) = ZERO
C      P = ZERO
C      H = X(2)-X(1)
C      IF (H .LE. ZERO) GO TO 75
C      F2 = -SM
C      FF = (F(2)-F(1))/H
C      IF (NX .LT. 3) GO TO 30
C      DO 5 I=3,NX
C      G = H
C      H = X(I)-X(I-1)
C      IF (H .LE. ZERO) GO TO 75
C      ONEDH = ONE/H
C      E = FF
C      FF = (F(I)-F(I-1))*ONEDH
C      Y(I) = FF-E
C      WK(4,I) = (G+H)*TWOD3

```

```

      WK(5,1) = H*Q*ED3
      ( (1,3,1) = DF(1-2)/G
        *X(1,1) = DF(1)-ONEDH
        WK(2,1) = -DF(1-1)/G-DF(1-1)*ONEDH
      5 CONTINUE
      DO 10 I=3,NX
        C(1-1,1) = WK(1,1)*WK(1,1)+WK(2,1)*WK(3,1)+WK(3,1)*WK(3,1)
        C(1-1,2) = WK(1,1)*WK(2,1)+WK(2,1)*WK(3,1)+1
        C(1-1,3) = WK(1,1)*WK(3,1)+2
      10 CONTINUE
      C
      15 IF (NX .LT. 3) GO TO 30
      DO 20 I=3,NX
        WK(2,1-1) = FF*WK(1,1-1)
        WK(3,1-2) = G*WK(1,1-2)
        WK(1,1) = ONE - (P*C(1-1,1)+WK(4,1)-FF*WK(2,1-1)-G*WK(3,1-2))
        WK(6,1) = Y(1)-WK(2,1-1)+WK(6,1-1)-WK(3,1-2)*WK(6,1-2)
        FF = P*C(1-1,2)+WK(5,1)+H*WK(2,1-1)
        G = H
        H = C(1-1,3)*P
      20 CONTINUE
      NP3 = NX+3
      DO 25 I=3,NX
        J = NP3-I
        WK(6,J) = WK(1,J)*WK(6,J)-WK(2,J)*WK(6,J+1)-WK(3,J)*WK(6,J+2)
      25 CONTINUE
      30 E = ZERO
      H = ZERO
      C
      DO 35 I=2,NX
        G = H
        H = (WK(6,1+1)-WK(6,1))/(X(1)-X(1-1))
        HMG = H-G
        WK(7,1) = HMG-DF(1-1)*DF(1-1)
        E = E+WK(7,1)*HMG
      35 CONTINUE
      G = -H*DF(NX)-DF(NX)
      WK(7,NP1) = G
      E = E-G*H
      G = F2
      F2 = E*P*P
      IF (F2 .GE. SM .OR. F2 .LE. G) GO TO 50
      FF = ZERO
      H = (WK(7,3)-WK(7,2))/(X(2)-X(1))
      IF (NX .LT. 3) GO TO 45
      DO 40 I=3,NX
        G = H
        H = (WK(7,1+1)-WK(7,1))/(X(1)-X(1-1))
        G = H-G-WK(2,1-1)*WK(1,1-1)-WK(3,1-2)*WK(1,1-2)
        FF = FF+G*WK(1,1)*G
        WK(1,1) = G
      40 CONTINUE
      45 H = E*P*FF
      IF (H .LE. ZERO) GO TO 50
      C
      C
      P = P+(SM-F2)/((SQRT(SM/E)+P)*H)
      GO TO 15
      C
      C
      50 NP1 = NX-1
      DO 55 I=1,NP1
        Y(I) = F(1)-P*WK(7,I+1)
        C(1,2) = WK(6,I+1)
        WK(1,1) = Y(I)
      55 CONTINUE

```



```

C      S(ROUTINE UERTST (IER,NAME)
C      INTEGER IER,NAME
C      SPECIFICATIONS FOR ARGUMENTS (
C      SPECIFICATIONS FOR LOCAL VARIABLES
C      INTEGER NAMEQ,NAMEQ
C      DATA NAMEQ/6H
C      DATA LEVEL/4, IEQDF/0, IEQ/1H-
C      IF (IER.GT.999) GO TO 25
C      IF (IER.LT.-32) GO TO 55
C      IF (IER.LE.128) GO TO 5
C      IF (LEVEL.LT.1) GO TO 30
C      CALL UGETIO(1,NIN,IOUNIT)
C      IF (IEQDF.EQ.1) WRITE(IOUNIT,35) IER,NAMEQ,IEQ,NAME
C      IF (IEQDF.EQ.0) WRITE(IOUNIT,35) IER,NAME
C      GO TO 30
C      5 IF (IER.LE.64) GO TO 10
C      IF (LEVEL.LT.2) GO TO 30
C      CALL UGETIO(1,NIN,IOUNIT)
C      IF (IEQDF.EQ.1) WRITE(IOUNIT,40) IER,NAMEQ,IEQ,NAME
C      IF (IEQDF.EQ.0) WRITE(IOUNIT,40) IER,NAME
C      GO TO 30
C      10 IF (IER.LE.32) GO TO 15
C      IF (LEVEL.LT.3) GO TO 30
C      CALL UGETIO(1,NIN,IOUNIT)
C      IF (IEQDF.EQ.1) WRITE(IOUNIT,45) IER,NAMEQ,IEQ,NAME
C      IF (IEQDF.EQ.0) WRITE(IOUNIT,45) IER,NAME
C      GO TO 30
C      15 CONTINUE
C      IF (NAME.NE.NAMEQ) GO TO 25
C      LEVEL = LEVEL
C      LEVEL = IER
C      IER = LEVEL
C      IF (LEVEL.LT.0) LEVEL = 4
C      IF (LEVEL.GT.4) LEVEL = 4
C      GO TO 30
C      25 CONTINUE
C      IF (LEVEL.LT.4) GO TO 30
C      PRINT NON-DEFINED MESSAGE
C      CALL UGETIO(1,NIN,IOUNIT)
C      IF (IEQDF.EQ.1) WRITE(IOUNIT,50) IER,NAMEQ,IEQ,NAME
C      IF (IEQDF.EQ.0) WRITE(IOUNIT,50) IER,NAME
C      RETURN
C      30 IEQDF = 0
C      35 FORMAT(19H *** TERMINAL ERROR,10X,7H( IER = ,13,
C      20H) FROM IMSL ROUTINE ,A6,A1,A6)
C      40 FORMAT(36H *** WARNING WITH FIX ERROR (IER = ,13,
C      20H) FROM IMSL ROUTINE ,A6,A1,A6)
C      45 FORMAT(18H *** WARNING ERROR,11X,7H( IER = ,13,
C      20H) FROM IMSL ROUTINE ,A6,A1,A6)
C      50 FORMAT(20H *** UNDEFINED ERROR,9X,7H( IER = ,15,
C      20H) FROM IMSL ROUTINE ,A6,A1,A6)
C      SAVE P FOR P = R CASE
C      P IS THE PAGE NAME
C      R IS THE ROUTINE NAME
C      55 IEQDF = 1
C      NAMEQ = NAME
C      65 RETURN
C      END
C      IMSL ROUTINE NAME - UGETIO
C

```

```

UERT0500
UERT0510
UERT0520
UERT0530
UERT0540
UERT0550
UERT0560
UERT0570
UERT0580
UERT0590
UERT0600
UERT0610
UERT0620
UERT0630
UERT0640
UERT0650
UERT0660
UERT0670
UERT0680
UERT0690
UERT0700
UERT0710
UERT0720
UERT0730
UERT0740
UERT0750
UERT0760
UERT0770
UERT0780
UERT0790
UERT0800
UERT0810
UERT0820
UERT0830
UERT0840
UERT0850
UERT0860
UERT0870
UERT0880
UERT0890
UERT0900
UERT0910
UERT0920
UERT0930
UERT0940
UERT0950
UERT0960
UERT0970
UERT0980
UERT0990
UERT1000
UERT1010
UERT1020
UERT1030
UERT1040
UERT1050
UERT1060
UERT1070
UERT1080
UERT1090
UERT1100
UERT1110
UERT1120
UERT1130
UGET0010
UGET0020

```


IT IS DESIGNED TO EVALUATE THE DEFINITE INTEGRAL OF A CONTINUOUS FUNCTION WITH FINITE LIMITS OF INTEGRATION. (REF.: L.F. SHAMPINE AND R.C. ALLEN, JR., NUMERICAL COMPUTING: AN INTRODUCTION, W.B. SAUNDERS CO., PP. 238-242 (1974)

F - NAME OF FUNCTION WHOSE INTEGRAL IS DESIRED. THE FUNCTION NAME F MUST APPEAR IN AN EXTERNAL STATEMENT IN THE CALLING PROGRAM.

A,B - LOWER AND UPPER LIMITS OF INTEGRATION.

ANS - APPROXIMATE VALUE OF THE INTEGRAL OF F(X) FROM A TO B.

AREA - APPROXIMATE VALUE OF THE INTEGRAL OF ABS(F(X)) FROM A TO B.

ERROR - ESTIMATED ERROR OF ANS. USER MAY WISH TO EXTRAPOLATE BY FORMING ANS+ERROR TO GET WHAT IS OFTEN A MORE ACCURATE RESULT, BUT NOT ALWAYS.

ACC - DESIRED ACCURACY OF ANS. CODE TRIES TO MAKE ABS(ERROR).LE.ACC*ABS(AREA).

IFLAG = 1 FOR NORMAL RETURN.

2 IF IT IS NECESSARY TO GO TO 30 LEVELS OR USE A SUBINTERVAL TOO SMALL FOR MACHINE WORD LENGTH, ERROR MAY BE UNRELIABLE IN THIS CASE.

3 IF MORE THAN 2000 FUNCTION EVALUATIONS ARE USED. ROUGH APPROXIMATIONS ARE USED TO COMPLETE THE COMPUTATIONS AND ERROR IS USUALLY UNRELIABLE.

DIMENSION FV(5),LORR(30),FIT(30),F2T(30),F3T(30),DAT(30),AREST(30),ESTT(30),EPST(30),PSUM(30)

SET U TO APPROXIMATELY THE UNIT ROUND-OFF OF SPECIFIC MACHINE (HERE IBM 360/67)

U=1.E-12

INITIALIZE

FOURU=4.0*U

IFLAG=1

EPS=ACC

ERROR=0.0

LVL=1

LORR(LVL)=1

PSUM(LVL)=0.0

ALPHA=A

DA=B-A

AREA=0.0

AREST=0.0

FV(1)=F(ALPHA)

FV(3)=F(ALPHA+0.5*DA)

FV(5)=F(ALPHA+DA)

KOUNT=3

WT=DA/6.0

EST=WT*(FV(1)+4.0*FV(3)+FV(5))

'BASIC STEP'. HAVE ESTIMATE EST OF INTEGRAL ON (ALPHA,ALPHA+DA). BISECT AND COMPUTE ESTIMATES ON LEFT AND RIGHT HALF INTERVALS. SIMILARLY TREAT INTEGRAL OF ABS(F(X)). SUM IS BETTER VALUE FOR INTEGRAL AND DIFF/15.0 IS APPROXIMATELY ITS ERROR.

DX=0.5*DA

FV(2)=F(ALPHA+0.5*DX)

FV(4)=F(ALPHA+1.5*DX)

KOUNT=KOUNT+2

```

W1=6.0
E1=WT*(FV(1)+4.0*FV(2)+FV(3))
E2=WT*(FV(3)+4.0*FV(4)+FV(5))
SUM=ESTL+ESTR
ARESTL=WT*(ABS(FV(1))+ABS(4.0*FV(2))+ABS(FV(3)))
ARESTR=WT*(ABS(FV(3))+ABS(4.0*FV(4))+ABS(FV(5)))
AREA=AREA1+(ARESTL+ARESTR)-AREST
DIFF=EST-SUM
C
C IF ERROR IS ACCEPTABLE. GO TO 2. IF INTERVAL IS TOO SMALL OR
C TOO MANY LEVELS OR TOO MANY FUNCTION EVALUATIONS, SET A FLAG
C AND GO TO 2 ANYWAY.
C
IF(ABS(DIFF).LE.EPS*ABS(AREA)) GO TO 2
IF(ABS(DX).LE.FOUR*ABS(ALPHA)) GO TO 5
IF(LVL.GE.30) GO TO 5
IF(MOUNT.GE.2000) GO TO 6
C
C HERE TO RAISE LEVEL. STORE INFORMATION TO PROCESS RIGHT HALF
C INTERVAL LATER. INITIALIZE FOR 'BASIC STEP' SO AS TO TREAT
C LEFT HALF INTERVAL.
C
LVL=LVL+1
LORR(LVL)=0
FIT(LVL)=FV(3)
F2T(LVL)=FV(4)
F3T(LVL)=FV(5)
DA=DX
DAT(LVL)=DX
ARESTL=ARESTL
AREST(LVL)=ARESTR
EST=ESTL
ESTT(LVL)=ESTR
EPS=EPS/1.4
EPST(LVL)=EPS
FV(5)=FV(3)
FV(3)=FV(2)
GO TO 1
C
C ACCEPT APPROXIMATE INTEGRAL SUM. IF IT WAS ON A LEFT INTERVAL.
C GO TO 'MOVE RIGHT'. IF A RIGHT INTERVAL, ADD RESULTS TO FIN-
C ISH AT THIS LEVEL. ARRAY LORR (MNEMONIC FOR LEFT OR RIGHT)
C TELLS WHETHER LEFT OR RIGHT INTERVAL AT EACH LEVEL.
C
2
3
ERROR=ERROR+DIFF/15.0
IF(LORR(LVL).EQ.0) GO TO 4
SUM=PSUM(LVL)+SUM
LVL=LVL-1
IF(LVL.GT.1) GO TO 3
ANS=SUM
RETURN
C
C 'MOVE RIGHT'. RESTORE SAVED INFORMATION TO PROCESS RIGHT HALF
C INTERVAL.
C
4
PSUM(LVL)=SUM
LORR(LVL)=1
ALPHA=ALPHA+DA
DA=DAT(LVL)
FV(1)=FIT(LVL)
FV(3)=F2T(LVL)
FV(5)=F3T(LVL)
ARESTL=ARESTT(LVL)
EST=ESTT(LVL)
EPS=EPST(LVL)
GO TO 1

```

```

C C A PT 'POOR' VALUE. SET APPROPRIATE FLAGS.
C C
C C 5 IFLAG=2
C C GO TO 2
C C 6 IFLAG=3
C C GO TO 2
C C END.
C C FUNCTION F1(X)
C C COMMON X,C,Y,NX
C C DIMENSION X(100),Y(100),C(99,3)
C C DO 1 I=2,NX
C C 1 IF(XX.LT.X(I))GO TO 2
C C 2 D=XX-X(I-1)
C C I=I-1
C C F1=((C(I,3)+D*(C(I,2))-D*(C(I,1)))*D*(Y(I)
C C RETURN
C C END.
C C SUBROUTINE DEPOSIT(CURR,EDEPOS,TO,Z,AW,MODE,DEPTH)
C C DEPOSIT IS TABATA'S ALGORITHM MODIFIED TO CALCULATE FAST ELECTRON CURRENTS AS
C C WELL AS DOSE RATES UNDER ELECTRON IRRADIATIONS. SEE TEXT.
C C
C C MODIFICATIONS MADE BY A.R. FREDERICKSON OF RADC. FOR ORIGINAL
C C ALGORITHM, SEE REF.22, VOL.1, PAGE 265.
C C
C C PURPOSE
C C CALCULATE THE ENERGY DEPOSITION OF ELECTRONS NORMALLY INCIDENT
C C ON THE SURFACE
C C
C C USAGE
C C EDEP = EDEPOS(DEPTH,TO,Z,AW,MODE)
C C
C C DESCRIPTION OF PARAMETERS
C C DEPTH - DEPTH FROM THE INCIDENT SURFACE EXPRESSED IN G/CM2
C C TO - INCIDENT KINETIC ENERGY OF THE ELECTRON IN MEV
C C Z - ATOMIC NUMBER OF THE ABSORBER
C C AW - ATOMIC WEIGHT OF THE ABSORBER
C C MODE - SHOULD BE SET AS FOLLOWS
C C =1 IN THE 1ST USAGE FOR A GIVEN SET OF
C C VALUES OF TO, Z, AND AW
C C GREATER THAN 2 IN THE 2ND AND LATER USAGES FOR THE
C C GIVEN SET OF VALUES OF TO, Z AND AW
C C EDEPOS - ENERGY DEPOSITION IN MEV/(G/CM2)
C C
C C STATEMENT FUNCTIONS FOR THE INVERSE EXTRAPOLATED RANGE-ENERGY
C C RELATION AND ITS DERIVATIVE
C C
C C D1(L,Z)=1470./Z**0.691*L**C3
C C D2(L,Z)=1.+D1(L,Z)
C C D3(L,Z)=(257.-0.34*Z)/D2(L,Z)
C C D4(L,Z,AW)=EXP(L*(C2+D3(L,Z)))/C1
C C T(L,Z,AW)=C1*(D4(L,Z,AW)-1.)
C C DIDX(L,Z,AW)=D4(L,Z,AW)*(C2+D3(L,Z))*(1.-C3*D1(L,Z)/D2(L,Z))
C C
C C CALCULATE DEPTH-INDEPENDENT TERMS
C C
C C IF(MODE=1) 7,7,27
C C 7 C1=2980./Z
C C C2=6.14*Z**1.026/AW
C C C3=0.505/Z**0.1874
C C C2=0.000178*Z
C C FK1=1.277/Z**0.134
C C TAU=TO/0.511004
C C BAK=1.15*EXP(-8.35/Z**0.925)/(1.+(.0185+15.7/Z**1.59)*TAU**0.156

```



```

1      -4.42/Z))
1      F      -1.33/Z+0.385*TAU+((0.086*Z+0.33)-0.048/Z+0.35/1) (0.47
1      -Z+0.12)
1      IF (FEB) 14,14,16
14      FB=0.
15      GO TO 17
16      FB=BAK*FEB
17      FRZTAU=C.000236*Z*TAU
18      REX=0.2335*AW/Z+1.209*(ALOG(1.+CZ*TAU)/CZ-(0.9891-0.000301*Z)*TAU
19      /11.+11.468-0.0118*Z)*TAU+((1.232/Z+0.109)))
1      FACTOR=(1.-FB*FRZTAU/(1.+FRZTAU))*70/1 (REX,Z,AW)
18      BETA=4.09/Z+0.28*TAU+((0.106*Z+0.103)
19      ALPHA=(1.-1./BETA)*((1.-BETA)
20      XRLIM=0.1*REX
21      W1=T(XRLIM,Z,AW)
22      A1=DTDX(XRLIM,Z,AW)/W1

C      CALCULATE DEPTH-DEPENDENT TERMS
C
27      IF (DEPTH) 28,30,30
28      EDEPOS=0.
29      CURR=0.
30      RETURN
31      XRESID=REX-FX1*DEPTH
32      XDEP=XRLIM-XRESID
33      IF (XDEP) 33,33,37
34      DMDX=DTDX(XRESID,Z,AW)*FX1
35      IF (DEPTH.EQ.0.) GO TO 43
36      A1=(XRESID,Z,AW)
37      GO TO 39
38      W1=EXP(-A1*XDEP)
39      DMDX=A1*FX1*W
40      S=DEPTH/REX
41      ASB=ALPHA*S+0.5*BETA
42      EDEPOS=FACTOR*EXP(-ASB)*(ASB*BETA/S/REX+W+DMDX)
43      CURR=(1.-BAK)*EXP(-ASB)
44      RETURN
45      END

```


*MISSION
of
Rome Air Development Center*

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C³I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

END

FILMED

5-84

DTIC