PULSE
An IBM 7094 Program for Calculation of Fast Neutron Kinetics by MonteCarlo

Addendum No.1, May 1964

A.E. Profio

Massachusetts Institute of Technology
Department of Nuclear Engineering
Cambridge, Massachusetts
PULSE

An IBM 7094 Program for Calculation of Fast Neutron Kinetics by MonteCarlo

Addendum No. 1, May 1964

A. E. Profio

Massachusetts Institute of Technology
Department of Nuclear Engineering
The purpose of this addendum is to bring up to date the report MITNE-40, "PULSE - An IBM 7094 Program for Calculation of Fast Neutron Kinetics by Monte Carlo", by A. E. Profio, issued in October 1963 by the Department of Nuclear Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts. Familiarity with the report is assumed.

The major change which has been made in the FORTRAN program is an improved method of treating neutron leakage. In the earlier version the distance and time to the next collision was computed in subroutine FLIGHT, and then one of the geometry subroutines BLOCK, CYLIND, or SPHERE was used to calculate the new position. The geometry subroutine also decided if the new position was inside or outside of the assembly. If it was outside, the neutron was tabulated as having escaped, but the time of escape was recorded at the time at which the next collision would have occurred, had the boundary not been crossed. This is evidently incorrect and biases the decay toward a longer mean lifetime.

In the new version, the distances to the various boundary planes (for a block), end planes or cylindrical surface (for a cylinder) or spherical boundary is calculated. There may be more than one positive distance to a boundary along the direction of travel of the neutron because the surfaces are supposed to be extended to infinity, and there is no way to know beforehand which boundary will be crossed first. Hence the distances to all boundaries are calculated along the direction of travel, and the smallest positive distance chosen as the distance to the surface of escape.

Outline of Revised Calculation

The main program of PULSE has been modified by substituting new instructions from FORTRAN statement number 130 up to but not including statement number 160, and by replacing subroutines FLIGHT, BLOCK, CYLIND, and SPHERE by new subroutines FLITE, POST,
DTPB, DTCB, and DTSB (see the listing included with this Addendum). The remainder of the program remains unchanged. The logical flow is now as follows: after returning from the cross section subroutine SIGMA and making a few checks as before, subroutine FLITE is called. FLITE is similar to FLIGHT, except that now only the time increment (flight time between collisions, not total time since start) is calculated as the variable TIMET. The main program then calls the appropriate subroutine to calculate distance to boundaries: DTPB (distance to plane boundary, for a block), DTCB (distance to the boundaries of a cylinder), or DTSB (distance to the sphere boundary). The subroutine to use is indicated as before by specification of the input variable KAS = 1 (block), 2 (cylinder), or 3 (sphere). The distance-to-boundary subroutines calculate the distances as explained above and find the smallest positive distance along the line of travel. The program then returns to (main) and calls subroutine PØST. This subroutine first compares the distance to the boundary, DISTB, with the distance to the next collision DIST which was previously computed by FLITE. If the distance to the boundary is larger than the collision distance, then the new position is calculated, the time is updated by TIMET, and calculations proceed since the neutron has not left the assembly. On the other hand, if the predicted collision distance is larger than the distance to the boundary, the new position is calculated (which should then be on the boundary; this information is not retained now but the main program could be rewritten to save it), and the time is updated to the time the neutron crosses the boundary. The leakage tabulation subroutine LEKTAL is then called as before.

New Subroutines

Subroutine FLITE is very similar to FLIGHT, the difference being that a time increment is returned rather than the current time plus the increment.

Subroutine PØST finds if DISTB is larger or smaller than DIST, calculates new positions as before, new times from either TIME + TIMET or TIME + DISTB/VEL, and returns also the index
variable KGEØ = 1 (if inside) or KGEØ = 2 (if escaping).

Subroutine DTPB computes the distance from the last collision at x, y, z to all six boundaries of the block, where the planes are assumed to be extended indefinitely. The distance is calculated along the direction of travel (specified by the direction cosines); negative distances indicate an intersection along the travel line but for the opposite direction in which the neutron is actually moving. The calculation is based on a vector formula given by Clark and Hansen (2253 Class Notes, M.I.T.):

\[ L = \frac{(r' - r) \cdot n}{\Omega \cdot n} \] (1)

where

- \( L \) = distance to plane
- \( r' \) = vector from arbitrary reference point to plane
- \( r \) = vector from reference point to neutron
- \( n \) = unit vector perpendicular to plane
- \( \Omega \) = unit vector in direction of travel of neutron

The reference point was taken at the origin of coordinates (center of block), \( r' \) and \( n \) are colinear, and \( r' \) is \( \pm XMAX, \pm YMAX, \) or \( \pm ZMAX \) depending on the plane being considered. \( \Omega = \alpha \hat{i} + \beta \hat{j} + \gamma \hat{k} \) where \( \hat{i}, \hat{j}, \hat{k} \) are the usual coordinate unit vectors. Substitution and simplification then gives the formulas in the FORTRAN listing.

The distance to the plane boundaries of a cylinder are calculated in DTCB as in Eq. 1. The distance to the (infinite) cylindrical surface is given by

\[
L = \frac{(n \cdot r)(n \cdot \Omega) - (r \cdot \Omega) + \left\{ \left[ (n \cdot r)(n \cdot \Omega) - (r \cdot \Omega) \right]^2 + [1 - (n \cdot \Omega)^2] \left[ \frac{R^2 + (n \cdot r)^2 - r^2}{1 - (n \cdot \Omega)^2} \right] \right\}^{1/2}}{1 - (n \cdot \Omega)^2}
\]
where \( L \) is the distance desired

\[ r = \text{vector from origin to the neutron} \]

\[ \hat{n} = \text{unit vector along axis of cylinder} \]

\[ \Omega = \text{unit vector in direction of travel} \]

\[ R = \text{cylinder radius} \]

\[ r = |r| = (x^2 + y^2 + z^2)^{1/2} \]

Subroutine DTSB calculates the distance to the spherical boundary using the relationship

\[ L = -\Omega \cdot r + \left\{ (\Omega \cdot r)^2 + r'^2 - r^2 \right\}^{1/2} \]

where \( L \) is the distance desired

\[ r = \text{vector from origin to the neutron} \]

\[ \Omega = \text{unit vector in direction of travel} \]

\[ r' = \text{sphere radius} \]

\[ r = |r| = (x^2 + y^2 + z^2)^{1/2} \]

Reduction of these equations gives the formulas in the FORTRAN listing.
*FORTRAN

LIST

DIMENSION SP(10), SBE1(20), SBI1(20), SBF1(20), SBC1(20), SBE2(20), SBI2(20), SBF2(20), SBC2(20), VB003D(20), AP1(10,20), AP2(10,20), NELS(100,10), NINS(100), KAPT(100)

READ 1, XSYSZ, SP, PARA, TP, KS, NEUT

1 FORMAT(7F8.4, I2, I14)

PRINT 2, XSYSZ, PARA, TP, KS, NEUT

2 FORMAT(1H1, 3HXS=F8.4, 2X, 3HY=F8.4, 2X, 3HZ=F8.4, 2X, 5HPARA=F8.4, 2X, 5HKS=F8.4, 2X, 5HNEUT=F8.4)

READ 3, SP

3 FORMAT(10F7.4)

PRINT 4, SP

4 FORMAT(10F7.4)

READ 5, XMAX, VMAX, RMAX, KAS

5 FORMAT(4F8.4)

PRINT 6, XMAX, VMAX, RMAX, KAS

6 FORMAT(10F7.4)

READ 7, TD, TCH, EMIN, ECH, KT1, KT2

7 FORMAT(4F7.3, 2I3)

PRINT 8, TD, TCH, EMIN, ECH, KT1, KT2

8 FORMAT(10F7.4, 2I2)

READ 9, P

9 FORMAT(11F6.2)

PRINT 10, P

10 FORMAT(10F7.4)

READ 11, VBOUND

11 FORMAT(10F7.4)

PRINT 12, VBOUND

12 FORMAT(10F7.4, 2I2)

READ 13, AD1, AL1, SLIM1, CIN1, VST1, FNU1, DELNU1, KIA1

13 FORMAT(7F8.4, 2F7.2, 2F8.4, 2I2)

PRINT 14, AD1, AL1, SLIM1, CIN1, VST1, FNU1, DELNU1, KIA1

14 FORMAT(10F7.4)

READ 15, HAD1, F7.5, 2X, 3HAI = F7.5, 2X, 6HAL1 = F7.5, 2X, 6HSL1 = F8.4, 2X, 7HK1 = F8.4, 2X, 7HK2 = F8.4, 2X, 7HK3 = F8.4, 2X, 7HK4 = F8.4)

15 FORMAT(10F7.4)

READ 16, J = 1, 20

16 FORMAT(10F7.4)

DO 17 J = 1, 20

17 FORMAT(10F7.4)

READ 18, SBE1(J)

18 FORMAT(10F7.4)

DO 19 J = 1, 20

19 FORMAT(10F7.4)

READ 20, SBI1(J)

20 FORMAT(10F7.4)

DO 21 J = 1, 20

21 FORMAT(10F7.4)

READ 22, SBF1(J)

22 FORMAT(10F7.4)

DO 23 J = 1, 20

23 FORMAT(10F7.4)
```
23  FORMAT(1HO,4HSBC=10F7.3/5X,10F7.3)
READ 15,AP1
PRINT 24,AP1

24  FORMAT(1HO,3HAP=10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,
110F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,
2F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X)
READ 15,VL1
PRINT 25,VL1

25  FORMAT(1HO,3HVL=10F7.3/4X,10F7.3)
READ 15,SBL1
PRINT 26,SBL1

26  FORMAT(1HO,4HSBL=10F7.3/(5X,10F7.3))
READ 27,FP1
PRINT 28,FP1

27  FORMAT(1IF6.3)
READ 29,FP1
PRINT 30,FP1

28  FORMAT(1HO,3HFPL=11F6.3/4X,11F6.3)
READ '3',AD2,A2,SLM2,CN2,VST2,FNU2,DELNU2,KIA2
PRINT 29,AD2,A2,SLM2,CN2,VST2,FNU2,DELNU2,KIA2

29  FORMAT(1HA4,4HAD2=F7.5,2X,3HA2=F7.5,2X,6HALIM2=F7.5,2X,6HLIM2=F8.4,
12X,5HCN2=F8.4,2X,5HVST2=F8.4,2X,5HFN2=F8.4,2X,7HDELNU2=F8.4,2X,
25HIA2=F8.4)
IF(AD2)40,40,30

30  IF(AD2)40,40,30
READ 15,SBET2
DO 31 J=1,20
31  SBET2(J)=AD2*SBET2(J)
PRINT 17,SBET2
READ 15,SB12
DO 32 J=1,20
32  SB12(J)=AD2*SB12(J)
PRINT 19,SB12
READ 15,SBF2
DO 33 J=1,20
33  SBF2(J)=AD2*SBF2(J)
PRINT 21,SBF2
READ 15,SB2
DO 34 J=1,20
34  SB2(J)=AD2*SB2(J)
PRINT 23,SB2
READ 15,AP2
PRINT 24,AP2
READ 15,VL2
PRINT 25,VL2
READ 15,SB2
PRINT 26,SB2
READ 27,FP2
PRINT 28,FP2
GO TO 50

40  DO 41 J=1,20
41  SBET2(J)=0.0
DO 42 J=1,20
42  SB12(J)=0.0
DO 44 J=1,20
44  SBF2(J)=0.0
DO 46 J=1,20
46  SB2(J)=0.0
DO 47 J=1,20
47  SB2(J)=0.0
50  REWIND KT1
REWIND KT2
KT=KT1
```
MULT=1
NL=0
NC=0
NS=0
NT=0
NF=0
NLTD=0
NGTR=0
NGZ=0
NLME=0
NGER=0
NOSL=0
KSCAT=0

100 DO 801 N=1,NEUT
110 CALL SOURCE(ALPHABETAGAMMAVELX,Y,Z,TIMEX,PARA,PARB,PARC,XT,YS,Z
15,ZMAX,THETA,SP,KS)
120 CALL SIGMA(VEL,SBE1,SBE2,SB11,SB12,SBF1,SBF2,SBC1,SBC2,AD1,AD2,VBO
1UND,TMFP,PE1,PE2,PI1,PI2,PF1,PF2,PC1,J)
IF(J)122,122,127
122 NT=NT+1
124 IF(NT-5)110,110,124
125 PRINT 125,NT
125 FORMAT(1HO93HNT=12)
GO TO 900
127 NT=0
130 CALL FLITE(DIST,TIMET,TMFP,VEL)
140 GOTO(145,150,155),KAS
145 CALL DTBP(ALPHABETAGAMMAXY,ZXMAX,2MAX,DIStB)
146 CALL POST(ALPHABETAGAMMAXY,Z,DIStB,TIMET,VEL,
1KGEO)
147 GOTO(160,600),KGEO
150 CALL DTCB(ALPHABETAGAMMA,X,Y,Z,RMAX,2MAX,DIStB)
151 GOTO146
155 CALL DTSB(ALPHABETAGAMMA,X,Y,Z,RMAX,DIStB)
156 GOTO146
160 IF(AD2)161,161,163
161 CALL COL1D1(PE1,PI1,PF1,KCOL)
GO TO 170
165 CALL COL1D2(PE1,PE2,PI1,PI2,PF1,PF2,PC1,KCOL)
170 KTYPE=KCOL/10
171 KNUCL=KCOL-(10*KTYPE)
GO TO (1200,300,400,500),KTYPE
200 CALL ELTAL(TIME,TDLCH,Z,ZMAX,KELSNELS)
NS=NS+1
GO TO (203,205,207,209),KELS
203 NLTD=NLTD+1
205 NGTR=NGTR+1
GO TO 209
207 NGZ=NGZ+1
GO TO 209
209 KSCAT=KSCAT+1
IF(KSCAT-100)211,211,225
211 GO TO (215,220),KNUCL
215 CALL ELSCAT(ALPHABETA,GAMMAC,VELA1,ALIM1,SLIM1,AP1,J)
GO TO 120
220 CALL ELSCAT(ALPHABETA,GAMMAC,VELA2,ALIM2,SLIM2,AP2,J)
GO TO 120
225 NOSL=NOSL+1
KSCAT=0
GO TO 800
300 CALL_INTAL (TIME, TD, TCH, KINS, NINS)
NS = NS + 1
GO TO (303, 305, 305, 305, 307) KINS
303 NLTD = NLTD + 1
GO TO 307
305 NGTR = NGTR + 1
GO TO 800
307 KSCAT = KSCAT + 1
IF(KSCAT - 100) 309, 309, 320
309 GO TO (310, 315) KNUCL
310 CALL_LEVEL(VEL, SBL1, VBOUND, PL, J)
CALL_INSIN(ALPHA, BETA, GAMMA, VEL, A1, CIN1, P, PL, VL1, VST1, KIA1)
GO TO 120
313 CALL_LEVEL(VEL, SBL2, VBOUND, PL, J)
CALL_INSIN(ALPHA, BETA, GAMMA, VEL, A2, CIN2, P, PL, VL2, VST2, KIA2)
320 NSL = NSL + 1
KSCAT = 0
GO TO 800
400 CALL_FISTAL(TIME, TD, TCH, KFIS, NFIS)
KSCAT = 0
GO TO (402, 404, 404, 404, 406) KFIS
402 NLTD = NLTD + 1
GO TO 406
404 NGTR = NGTR + 1
GO TO 800
406 GO TO (407, 409, 409) KNUCL
407 CALL_FISSN(X, Y, Z, VEL, TIME, FP1, FNU1, DELNU1, NF, KT)
GO TO 800
409 CALL_FISSN(X, Y, Z, VEL, TIME, FP2, FNU2, DELNU2, NF, KT)
GO TO 800
500 CALL_CAPTAL(TIME, TD, TCH, KCAP, KAPT)
NC = NC + 1
KSCAT = 0
GO TO (504, 506, 506, 506, 507) KCAP
504 NLTD = NLTD + 1
GO TO 800
506 NGTR = NGTR + 1
507 GO TO 800
600 CALL_MKETAL(TIME, VEL, TD, TCH, EMIN, ECH, KLEK, LEAK)
NL = NL + 1
KSCAT = 0
GO TO (604, 606, 608, 610, 611) KLEK
604 NLTD = NLTD + 1
GO TO 800
606 NGTR = NGTR + 1
GO TO 800
608 NLME = NLME + 1
GO TO 800
610 NGER = NGER + 1
611 GO TO 800
800 GO TO (801, 809, 820) MULT
801 CONTINUE
KS = 1
803 MULT = 2
REWIND KT1
REWIND KT2
IF (NF) 850, 807
807 N = NF
NF = 0
809 N = N - 1
IF (N) 814, 811
811 READ TAPE KT1 * XS * YS * ZS * PARA * THETA
KT = KT2
GO TO 110
814 MULT = 3
REWIND KT1
REWIND KT2
IF (NF) 850, 818
818 N = NF
NF = 0
820 N = N - 1
IF (N) 803, 822
822 READ TAPE KT2 * XS * YS * ZS * PARA * THETA
KT = KT1
GO TO 110
850 PRINT 851, NL, NC, NS, NF, NLTD, NGTR, NGZR, NLME, NGER, NOSL
851 FORMAT (1H1, 3HNL = 18, 2X, 3HNC = 18, 2X, 3HNS = 18, 2X, 3HNF = 18 / 1H0, 5HNLTD = 18, 12X, 5HNGTR = 18, 2X, 5HNGZR = 18, 2X, 5HNLME = 18, 2X, 5HNGER = 18, 2X, 5HNOZ = 18)
PRINT 853, LEAK
853 FORMAT (1H0, 5HLEAK = 2016 / (6X, 2016))
PRINT 855, NELS
855 FORMAT (1H4, 5HNELS = 2016 / (6X, 2016))
PRINT 857, NINS
857 FORMAT (1H4, 5HNINS = 2016 / (6X, 2016))
PRINT 859, NFIS
859 FORMAT (1H4, 5HNFIS = 2016 / (6X, 2016))
PRINT 861, KAPT
861 FORMAT (1H4, 5HKAPT = 2016 / (6X, 2016))
PUNCH 863, LEAK, NELS, NINS, NFIS, KAPT
863 FORMAT (10H)
900 CALL EXIT
END
SUBROUTINE SOURCE(ALPHA, BETA, GAMMA, VEL, X, Y, Z, TIME, PARA, PARB, PARC, X, Y, Z, MAX, THETA, #, KS)
DIMENSION SP(10)
GO TO (10, 20, 30, 40), KS
10 X = XS
   Y = YS
   Z = ZS
   GAMMAC = 2.0 * RANNOF(W) - 1.0
   VEL = PARA
   CALL ISOANG(ALPHA, BETA, GAMMA, GAMMAC, VEL)
   TIME = THETA
   RETURN
20 X = XS * (2.0 * RANNOF(V) - 1.0)
   Y = YS * (2.0 * RANNOF(W) - 1.0)
   Z = ZS
   GAMMA = 1.0
   ALPHA = 0.0
   BETA = 0.0
   VEL = PARA - PARB * RANNOF(U)
   TIME = 0.0
   RETURN
30 VEL = PARA - PARB * RANNOF(V) - PARC * (1.0 - GAMMAC)
   CALL ISOANG(ALPHA, BETA, GAMMA, GAMMAC, VEL)
   S = (-ZMAX - ZS) / GAMMA
   X = S * ALPHA
   Y = S * BETA
   Z = ZMAX
   TIME = S / VEL
   RETURN
40 CALL TARGET(ALPHA, BETA, GAMMA, VEL, X, Y, Z, TIME, PARA, PARB, PARC)
RETURN
END
SUBROUTINE ANGLS (SP, GAMMAC)
DIMENSION SP(10)
R=RANNOF(X)
M=10.0*R+1.0
REM=R-0.1*FLOATF(M-1)
IF(10-M)30,10,20
10 GAMMAC=SP(10)+(REM/0.1)*(1.0-SP(10))
RETURN
20 GAMMAC=SP(M)+(REM/0.1)*(SP(M+1)-SP(M))
RETURN
30 GAMMAC=1.0
RETURN
END
SUBROUTINE TARGET(ALPHA, BETA, GAMMA, VEL, X, Y, Z, TIME, PARA, PARB, PARC)
X=X
Y=Y
Z=Z
ALPHA=ALPHA
BETA=BETA
GAMMA=GAMMA
TIME=0.0
VEL=PARA-PARB*RANNOF(V)-PARC*ABSF(GAMMA)
RETURN
END
SUBROUTINE SIGMA(EN, SBE1, SBE2, SBI1, SBI2, SBF1, SBF2, SBC1, SBC2, AD1, AD2, EBOUND, TMFP, PE1, PE2, PI1, PI2, PF1, PF2, PC1, J)
DIMENSION SBE1(20), SBE2(20), SBI1(20), SBI2(20), SBF1(20), SBF2(20), SBC1(20), SBC2(20), EBOUND(20)

10 CALL GROUP(EN, EBOUND, J, KGP)
  J = J
11 GO TO (12, 14, KGP)
12 J = 0
13 RETURN
14 IF(20 - J) 160, 60, 20
20 SE1 = FIND(EN, J, EBOUND, SBE1)
21 SI1 = FIND(EN, J, EBOUND, SBI1)
22 SF1 = FIND(EN, J, EBOUND, SBF1)
23 SC1 = FIND(EN, J, EBOUND, SBC1)
24 IF(AD2) 25, 25, 30
25 SE2 = 0.
26 SI2 = 0.
27 SF2 = 0.
28 SC2 = 0.
29 GO TO 40
30 SE2 = FIND(EN, J, EBOUND, SBE2)
31 SI2 = FIND(EN, J, EBOUND, SBI2)
32 SF2 = FIND(EN, J, EBOUND, SBF2)
33 SC2 = FIND(EN, J, EBOUND, SBC2)
40 TMFP = 1.0/(SE1 + SI1 + SF1 + SC1 + SE2 + SI2 + SF2 + SC2)
41 PE1 = TMFP * SE1
42 PI1 = TMFP * SI1
43 PF1 = TMFP * SF1
44 IF(AD2) 45, 45, 50
45 PC1 = 1.0 - PE1 - PI1 - PF1
46 IF(PC1 < 0.0001) 47, 48, 48
47 PC1 = 0.0
48 RETURN
50 PC1 = TMFP * SC1
51 PE2 = TMFP * SE2
52 PI2 = TMFP * SI2
53 PF2 = TMFP * SF2
54 RETURN
60 SE1 = SBE1(20)
61 SI1 = SBI1(20)
62 SF1 = SBF1(20)
63 SC1 = SBC1(20)
64 SE2 = SBE2(20)
65 SI2 = SBI2(20)
66 SF2 = SBF2(20)
67 SC2 = SBC2(20)
68 GO TO 40
END
SUBROUTINE GROUP(EN, EBOUND, J, KGP)
DIMENSION EBOUND(20)

10 IF(EN-EBOUND(1))11,13,13
11 KGP=1
12 RETURN
13 J=20
14 IF(EN-EBOUND(J))15,91,91
15 J=10
16 IF(EN-EBOUND(J))17,91,29
17 J=5
18 IF(EN-EBOUND(J))19,91,25
19 J=2
20 IF(EN-EBOUND(J))90,91,21
21 J=J+1
22 IF(EN-EBOUND(J))90,91,23
23 J=J+1
24 IF(EN-EBOUND(J))90,91,91
25 J=7
26 IF(EN-EBOUND(J))27,91,21
27 J=J-1
28 GO TO 24
29 J=15
30 IF(EN-EBOUND(J))31,91,33
31 J=12
32 IF(EN-EBOUND(J))27,91,21
33 J=17
34 IF(EN-EBOUND(J))27,91,21
90 J=J-1
91 KGP=2
92 RETURN
END
FUNCTION FIND(EN,J,EBOUND,SBX)
DIMENSION EBOUND(20),SBX(20)
FIND=SBX(J)+(EN-EBOUND(J))*(SBX(J+1)-SBX(J))/(EBOUND(J+1)-EBOUND(J))
RETURN
END
SUBROUTINE FLITE(DIST, TIME, TMFP, VEL)
10 B = RANNOF(X)
12 IF (B - 0.0000454) 10, 10, 13
13 C = LOGF(B)
15 DIST = TMFP * (-C)
16 TIME = DIST / VEL
18 RETURN
END
SUBROUTINE POST(ALPHA, BETA, GAMMA, X, Y, Z, DIST, DISTB, TIME, 
TIMET, VEL, KGE0)
IF(DISTB == DIST) 20 20 10
10 X = X + ALPHA * DISTB
Y = Y + BETA * DISTB
Z = Z + GAMMA * DISTB
TIME = TIME + DISTB/VEL
KGE0 = 2
RETURN
20 X = X + ALPHA * DIST
Y = Y + BETA * DIST
Z = Z + GAMMA * DIST
TIME = TIME + TIMET
KGE0 = 1
RETURN
END
SUBROUTINE DTPB (ALPHA, BETA, GAMMA, X, Y, Z, XMAX, YMAX, ZMAX, DISTB)
D1 = (XMAX - X) / ALPHA
D2 = -(XMAX + X) / ALPHA
D3 = (YMAX - Y) / BETA
D4 = -(YMAX + Y) / BETA
D5 = (ZMAX - Z) / GAMMA
D6 = -(ZMAX + Z) / GAMMA
IF (D1) 10, 11, 11
10 D1 = 10000.0
11 IF (D2) 12, 13, 13
12 D2 = 10000.0
13 IF (D3) 14, 15, 15
14 D3 = 10000.0
15 IF (D4) 16, 17, 17
16 D4 = 10000.0
17 IF (D5) 18, 19, 19
18 D5 = 10000.0
19 IF (D6) 20, 21, 21
20 D6 = 10000.0
21 DISTB = MIN(F(D1), D2, D3, D4, D5, D6)
22 IF (DISTB) 23, 24, 24
23 DISTB = 0.0
24 RETURN
END
SUBROUTINE DTCB (ALPHA, BETA, GAMMA, X, Y, Z, RMAX, ZMAX, DISTB)
OMR = X*ALPHA + Y*BETA + Z*GAMMA
R = SQRTF((X**2 + Y**2 + Z**2)
D1 = (Z*GAMMA - OMR + SQRTF((Z*GAMMA - OMR)**2 + (1.0 - GAMMA**2)*RMAX**2 + Z**2 - R**2)) / (1.0 - GAMMA**2)
IF (DISTB > 35) 35, 40
35 DISTB = -DISTB
40 RETURN
END
SUBROUTINE DTSB(ALPHA, BETA, GAMMA, X, Y, Z, RMAX, DISTB)
OMR = X*ALPHA + Y*BETA + Z*GAMMA
R = SQRTF(X**2 + Y**2 + Z**2)
DISTB = OMR + SQRTF(OMR**2 + RMAX**2 - R**2)
5 IF(DISTB)20,10,10
10 RETURN
20 DISTB = -DISTB
GO TO 5
END
SUBROUTINE LEKTAL(TIME, VEL, TD, TCH, EMIN, ECH, KLEK, LEAK)
  DIMENSION LEAK(100,10)
  ITIME=(TIME-TD)/TCH
  IF(100-ITIME)12,14,14
  KLEK=1
  RETURN
  IF(ITIME-1)12,14,14
  KLEK=2
  RETURN
  IEN=(0.5227*(VEL**2)-EMIN)/ECH
  IF(IEN-1)21,21,21
  KLEK=3
  RETURN
  KLEK=4
  RETURN
  LEAK(ITIME+IEN)=LEAK(ITIME+IEN)+1
  KLEK=5
  RETURN
END
SUBROUTINE COLID1(PE1, PI1, PF1, KCOL)
9  R = RANNOF(X)
10  IF (R - PE1) 20, 11, 11
11  IF (R - PE1 - PI1) 30, 12, 12
12  IF (R - PE1 - PI1 - PF1) 40, 13, 13
13  KCOL = 41
14  RETURN
20  KCOL = 11
21  RETURN
30  KCOL = 21
31  RETURN
40  KCOL = 31
41  RETURN
END
SUBROUTINE COLID2(PE1, PE2, PI1, PI2, PF1, PF2, PC1, KCOL)
9 R = RANNOF(X)
10 IF (R - PE1) 20, 11, 11
11 IF (R - PE1 - PE2) 30, 12, 12
12 IF (R - PE1 - PE2 - PI1) 40, 13, 13
13 IF (R - PE1 - PE2 - PI1 - PI2) 50, 14, 14
14 IF (R - PE1 - PE2 - PI1 - PI2 - PF1) 60, 15, 15
15 IF (R - PE1 - PE2 - PI1 - PI2 - PF1 - PF2) 70, 16, 16
16 IF (R - PE1 - PE2 - PI1 - PI2 - PF1 - PF2 - PC1) 80, 90, 90
20 KCOL = 11
21 RETURN
30 KCOL = 12
31 RETURN
40 KCOL = 21
41 RETURN
50 KCOL = 22
51 RETURN
60 KCOL = 31
61 RETURN
70 KCOL = 32
71 RETURN
80 KCOL = 41
81 RETURN
90 KCOL = 42
91 RETURN
END
SUBROUTINE ELTAL(TIME, TD, TCH, Z, ZMAX, KELS, NELS)
DIMENSION NELS(100,10)

ITIME = (TIME - TD) / TCH

IF(ITIME > 1) 12, 14, 14
KELS = 1
RETURN
12
IF(100 - ITIME) 15, 17; 17
KELS = 2
RETURN
15
IF(10 - ITIME) 22, 24, 24
KELS = 4
RETURN
22
IF((ITIME * TCH) / ZMAX) 19, 21
KELS = 3
RETURN
19
IF(ITIME * TCH) 23, 26
KELS = 5
RETURN
23
NELS((ITIME * TCH) + NELS) = NELS((ITIME * TCH) + 1)
RETURN
26
END
SUBROUTINE ELSCAT(ALPHA,BETA,GAMMA,VEL,ALIM,SLIM,AP,J)
DIMENSION AP(10,20)
10 IF(VEL-SLIM) 11,20,20
11 GAMMAC=2.0*RANNOF(X)-1.0
12 IF(A-ALIM)13,15,15
13 CALL CMLAB(ALPHA,BETA,GAMMA,GAMMAC,VEL)
14 RETURN
15 CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL)
16 RETURN
20 CALL ANGLE(J,AP,GAMMAC)
21 GO TO 12
END
SUBROUTINE ANGLE(J, AP, GAMMAC)
DIMENSION AP(10,20)
R=RANNOF(X)
M=10*R+1
REM=R-0.1*FLOAT(M-1)
IF(10-M)30,10,20
10 GAMMAC=AP(10,J)+(REM/0.1)*(1.0-AP(10,J))
RETURN
20 GAMMAC=AP(M,J)+(REM/0.1)*(AP(M+1,J)-AP(M,J))
RETURN
30 GAMMAC=1.0
RETURN
END
SUBROUTINE CMLAB(ALPHA,BETA,GAMMA,GAMMAC,VEL,A)
10 R1=RANNOF(X)
11 R2=RANNOF(X)
12 ETA=(2.0*R1-1.0)**2+(2.0*R2-1.0)**2
13 IF(ETA-1.0)14,10
14 ROOT=SQRTF(1.0-GAMMAC**2)/ETA
15 ALPHAC=(2.0*R1-1.0)*ROOT
16 BETAC=(2.0*R2-1.0)*ROOT
17 RTG=SQRTF(1.0-GAMMA**2)
18 ALPHAP=((ALPHA*GAMMA*ALPHAC-BETA*BETAC)/RTG)+ALPHA*GAMMAC
19 BETAP=((BETA*GAMMA*ALPHAC+ALPHA*BETAC)/RTG)+BETA*GAMMAC
20 GAMMAP=-ALPHAC*RTG+GAMMA*GAMMAC
21 RTA=SQRTF(1.0+A**2+2.0*A*GAMMAC)
22 ALPHA=(ALPHA+A*ALPHAP)/RTA
23 BETA=(BETA+A*BETAP)/RTA
24 GAMMA=(GAMMA+A*GAMMAP)/RTA
25 VEL=(VEL/RTA)/(A+1.0)
26 RETURN
END
SUBROUTINE ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL)
   R1=RANNOF(X)
   R2=RANNOF(X)
   ETA=(2.0*R1-1.0)**2+(2.0*R2-1.0)**2
   ROOT=SQRTF((1.0-GAMMA**2)/ETA)
   ALPHA=(2.0*R1-1.0)*ROOT
   BETA=(2.0*R2-1.0)*ROOT
   VEL=VEL
   RETURN
END
SUBROUTINE INTAL(TIME, TD, TCH, KINS, NINS)
DIMENSION NINS(100)
ITIME = (TIME - TD) / TCH
IF (ITIME - 1) 12, 14, 14
KINS = 1
RETURN
IF (100 - ITIME) 15, 17, 17
KINS = 2
RETURN
NINS(ITIME) = NINS(ITIME) + 1
KINS = 5
RETURN
END
SUBROUTINE LEVEL(VEL, SBL, VBOUND, PL, J)
DIMENSION SBL(20), VBOUND(20), PL(20), SL(20)
IF(20-J)10,10,20
10 doi L=1,20
SL(L)=SBL(L,20)
15 CONTINUE
GO TO 30
20 doi L=1,20
SL(L)=SBL(L,J)+(VEL-VBOUND(J))*(SBL(L,J+1)-SBL(L,J))/(VBOUND(J+1)-
1VBOUND(J))
25 CONTINUE
SUM=0.0
DO 30 L=1,20
30 SUM=SUM+SL(L)
CONTINUE
SUM=1.0/SUM
DO 35 L=1,20
PL(L)=SUM*SL(L)
35 CONTINUE
RETURN
END
SUBROUTINE INSCAT(ALPHA,BETA,GAMMA,VEL,A,CIN,P,PL,VL,VST,KIA)
DIMENSION PL(20),VL(20),P(22)
10 GO TO (11,14) KIA
11 GAMMAC=2*0*RANNOF(X)-1.0
12 CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL)
13 GO TO 20
14 CALL ANGLI(VEL,A,GAMMAC)
15 GO TO 12
20 IF(VEL-VST)21,30,30
21 R1=RANNOF(X)
22 L=1
23 SUM=0.
24 SUM=SUM+PL(L)
25 IF(R1-SUM)28,26,26
26 L=L+1
27 GO TO 24
28 VEL=VEL-VL(L)
29 RETURN
30 CALL INSPEC(VEL,CIN,P)
31 RETURN
END
SUBROUTINE ANGLI(VEL, A, GAMMAC)
GAMMAC = 1.0
VEL = VEL
A = A
RETURN
END
SUBROUTINE INSPEC(VEL, CIN, P)
DIMENSION P(22)
VMAX = CIN * VEL
R = RANNOF(X)
K = 20.0 * R + 1.0
REM = R - 0.05 * FLOAT(K - 1)
W = P(K) + (REM / 0.05) * (P(K + 1) - P(K))
VEL = W * VMAX
RETURN
END
SUBROUTINE CAPTAL(TIME,TD,TCH,KCAP,KAPT)
  DIMENSION KAPT(100)
  ITIME=(TIME-TD)/TCH
  IF(ITIME=1)12,14,14
  KCAP=1
  RETURN
  IF(100-ITIME)15,17,17
  KCAP=2
  RETURN
  KAPT(ITIME)=KAPT(ITIME)+1
  KCAP=5
  RETURN
END
SUBROUTINE FISTAL(TIME, TD, TCH, KFIS, NFIS)

dimension NFIS(100)

ITIME = (TIME - TD) / TCH

IF (ITIME - 1) 12, 14, 14

KFIS = 1
RETURN

IF (100 - ITIME) 15, 17, 17

KFIS = 2
RETURN

NFIS (ITIME) = NFIS (ITIME) + 1

KFIS = 5
RETURN

END
SUBROUTINE FISSN(X,Y,Z,VEL,TIME,FP,FNU,DELNU,NF,KT)

DIMENSION FP(22)

FISNO=FNU+DELNU*VEL**2

IF(FISNO<.0120,30,40)

R1=RANNOF(W)+2.0

IF(R1=FISNO)30,30,25

I=2

GO TO 50

30 I=3

IF(FISNO<.00120,49,49)

R2=RANNOF(W)+3.0

IF(R2=FISNO)49,49,45

45 I=3

GO TO 50

49 I=4

DO 60 N=1,I

R3=RANNOF(W)

K=20.0*R3+1.0

REM=R3-0.05*FLOATF(K-1)

PARA=FP(K)+(REM/0.05)*(FP(K+1)-FP(K))

THETA=TIME

XS=X

YS=Y

ZS=Z

WRITE TAPE KT,XS,YS,ZS,PARA,THETA

DO 60 N=NF+1

RETURN

END

TOTAL 795*