DR-1822-P

U.S. DEPARTMENT OF COMMERCE
National Technical Information Service

ORNL-4972

THE MORSE MONTE CARLO RADIATION TRANSPORT CODE SYSTEM

M. B. Emmett

Oak Ridge National Laboratory
Oak Ridge, Tennessee

February 1975

ORNL--4972

DE86 006924

Received by OSTI
FEB 25 1986

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

ORNL-4972

86006924

The MORSE Monte Carlo Radiation Transport Code System

M. B. Emmett

PRICES SUBJECT TO CHANGE

OAK RIDGE NATIONAL LABORATORY
OPERATED BY UNION CARBIDE CORPORATION FOR THE U.S. ATOMIC ENERGY COMMISSION

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
US Department of Commerce
Springfield, VA. 22151

ORNL-4972
UC-32 — Mathematics and Computers

Contract No. W-7405-eng-26

Neutron Physics Division

THE MORSE MONTE CARLO RADIATION TRANSPORT CODE SYSTEM

M. B. Emmett*

NOTE:
This Work Supported by
DEFENSE NUCLEAR AGENCY
Under Subtask PE 074

*Computer Sciences Division, UCC Nuclear Division.

FEBRUARY 1975

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
U.S. ATOMIC ENERGY COMMISSION

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
Price: Printed Copy $10.60; Microfiche $2.25

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

PART I

General Introduction to the MORSE Code System

TABLE OF CONTENTS

1.1 INTRODUCTION

The MORSE[1-4] Code System has been in continuous use for several years now and has subsequently undergone much revision. In addition to this, its development has continued; and its applications have expanded into new areas requiring development of additional computer programs. Because the documentation of the various modules of MORSE is scattered through several reports and because of the numerous revisions required to get this documentation up-to-date for the current versions of routines, it was decided to write one complete report describing all aspects of the MORSE system and related computer codes. As might be expected, this report draws heavily from the previous reports. All information necessary to use these codes is consolidated into one report. There are sections containing descriptions of the MORSE and PICTURE[5] codes, input descriptions, sample problems, derivations of the physical equations and explanations of the various error messages. This report attempts to document the IBM-360, UNIVAC-1108 and CDC-6600 versions of MORSE and its auxiliary codes. Most options are available on all three machines, but some are not because one or more of the machines does not have some particular capability. It is expected that some of the parts being published now will be expanded, in particular, the sample problem notebook in Part 3 will have additional examples to illustrate more of the options available. Additional parts such as a manual for DOMINO[6] and a description of the collision density fluence estimator[7] which is included in the basic MORSE on the UNIVAC machine will also be added. Because time was at a premium, no attempt was made to update flow charts and include them. Users may refer to the previous reports (see refs. 1-4 on page 1.2-1), but should keep in mind that some of the charts are out of date.

This report will describe only the MORSE with combinatorial geometry (CG) and not the 05R-type geometries which are no longer maintained in MORSE. This decision was made because CG will handle all types of geometries expeditiously and because it is quite cumbersome to maintain multiple versions of a code as large and complicated as MORSE.

It is worth noting at this point that the MORSE with combinatorial geometry does not always produce the same random number sequence when a job is restarted at some intermediate point. This results from the fact that this geometry package has a "memory" - i.e. it uses a table look-up rather than a re-calculation when the same path is encountered. All 05R-type geometries calculated the path each time and, therefore, repeated random number sequences.

The format of this documentation is such that updates, deletions and additions should be easily done. Each "part" is like a separate report because it contains its own table of contents and references and its pages are numbered beginning with 1.

1.2 REFERENCES

1. E. A. Straker, et al, "The MORSE Code - A Multigroup Neutron and Gamma-Ray Monte Carlo Transport Code," ORNL-4585 (1970).
2. E. A. Straker and M. B. Emmett, "MORSEC, A Revised Cross-Section Module for the MORSE Multigroup Monte Carlo Code," ORNL-4716 (1971).
3. V. R. Cain, "SAMBO - A Collision Analysis Package for Monte Carlo Codes," ORNL-TM-3203 (September 1970).
4. E. A. Straker, W. H. Scott, Jr., and N. R. Byrn, "The MORSE-Code with Combinatorial Geometry," DNA-2860T (May 1972).
5. D. C. Irving and G. W. Morrison, "PICTURE: An Aid in Debugging Geometry Input Data," ORNL-TM-2892 (May 1970).
6. M. B. Emmett, C. E. Burgart, and T. J. Hoffman, "DOMINO, A General Purpose Code for Coupling Discrete Ordinates and Monte Carlo Radiation Transport Calculations," ORNL-4853 (1973).
7. E. A. Straker and M. B. Emmett, "Collision Site Plotting Routines and Collision Density Fluence Estimates for the MORSE Monte Carlo Code," ORNL-TM-3585 (October 1971).

PART II

A Guide on How to Use the Various Codes
(to be published)

PART III

Sample Problem Notebook

TABLE OF CONTENTS

LIST OF TABLES

3.1 INTRODUCTION

In order to illustrate the use of the computer codes documented in other parts of this report, a set of sample problems is being assembled. These problems should give potential users some insight into the various applications of these codes as well as indicating how to set up certain types of problems. This part of the report will be supplemented with additional problems from time to time.

3.2 MORSE Sample Problem #1

The fast-neutron fluence at several radial distances is calculated by MORSE for a point, isotropic, fission source in an infinite medium of air. The air was assumed to be made up of only oxygen and nitrogen with a total density of 1.29 g/ℓ. The combinatorial geometry package (CG) was used to describe the concentric spherical shells of air surrounding the point source. Although the entire medium was air, the geometry medium numbers alternate between each of the shells for use with the boundary-crossing estimator.† This estimator requires that each detector lie on a boundary separating two media. The cross sections for air used in this calculation were for 22 neutron groups with five Legendre coefficients used for the angular expansion. Only the top 13 neutron groups were analyzed. The group structure with the corresponding fraction of particles emitted in each group is given in Table 3.1. Splitting, Russian roulette, and path length stretching were also implemented. Input data is listed in Table 3.2. The output listing follows.

For this problem, the standard SØURCE is used, and BANKR is modified to call SDATA and BDRYX during the particle walk. SDATA is a routine for analysis of uncollided fluence, and BDRYX is for analysis of all boundary crossings (equivalent to path length/unit volume). (See Part 4, Section 4.6.4)

†See Part 4, Section 4.5.3, pg. 4.5-28.

Table 3.1. Fission Spectrum in 14-Group Structure

Group No.	Energy Limits (MeV)	Fraction of Source Neutrons
1	15.0-12.21	1.5579(-4)[a]
2	12.21-10.0	8.9338(-4)
3	10.0-8.187	3.4786(-3)
4	8.187-6.36	1.3903(-2)
5	6.36-4.966	3.4557(-2)
6	4.966-4.066	3.5047(-2)
7	4.066-3.012	1.0724(-1)
8	3.012-2.466	8.8963(-2)
9	2.466-2.350	2.3186(-2)
10	2.350-1.827	1.2030(-1)
11	1.827-1.108	2.1803(-1)
12	1.108-0.5502	1.9837(-1)
13	0.5502-0.1111	1.4036(-1)
14	0.1111-0.3355	1.5489(-2)

[a]Read as 1.5579×10^{-4}.

Table 3.2. Listing of Input Cards for MØRSE Sample Problem #1

```
MORSE SAMPLE PROBLEM  POINT FISSION SOURCE IN AIR
  200  400    10     1   13     0   22   22     0     0     5     1     0          CARD B
    0   14     0    01.0          0.0          0.0          1.0          2.2     +5 CARD C
       0.0          0.0          0.0          0.0          0.0          0.0     0.0 CARD D
 1.5579 -4 8.9338 -4 3.4766 -3 1.3903 -2 3.4557 -2 3.5047 -2 1.0724 -1 CAEOS E
 8.8963 -2 2.3186 -2 1.2030 -1 2.1803 -1 1.9837 -1 1.4036 -1 1.5489 -2 CARDS E
 1.5000 +7 1.2214 +7 1.0000 +7 8.1873 +6 6.3600 +6 4.9658 +6 4.0657 +6 CARDS F
 3.0119 +6 2.4659 +6 2.3500 +6 1.8268 +6 1.1080 +6 5.5020 +5 1.1109 +5 CARDS F
 3.3546 +3 5.8294 +2 1.0130 +2 2.9020 +1 1.0677 +1 3.0590 +1 1.1253 +1 CARDS F
 4.1400 -1                                                                  CARDS F
J00035FA731A
    1    1    1    0    0    1   13                                         CARD I
    0    0    0    0    0    0 1.0   +01 1.0   -02 1.0   -01 5.0    -1 CARDS J
   -1
    0    0    0    0                                                       CARD L
    0    0          SAMPLE PROB. 1 FOR MORSE
  SPH            0.          0.          0.       3.0E +3
  SPH            0.          0.          0.       5.0E +3
  SPH            0.          0.          0.       7.5E +3
  SPH            0.          0.          0.       1.0E +4
  SPH            0.          0.          0.       1.5E +4
  SPH            0.          0.          0.       2.0E +4
  SPH            0.          0.          0.       3.0E +4
  SPH            0.          0.          0.       6.0E +4
  SPH            0.          0.          0.       7.0E +4
  SPH            0.          0.          0.       9.0E +4
  SPH            0.          0.          0.       1.2E +5
  SPH            0.          0.          0.       1.5E +5
  SPH            0.          0.          0.       1.0E +6
  SPH            0.          0.          0.       1.0E +7
  END
  AIR           +1
  AIR           +2         -1
  AIR           +3         -2
  AIR           +4         -3
  AIR           +5         -4
  AIR           +6         -5
  AIR           +7         -6
  AIR           +8         -7
  AIR           +9         -8
  AIR          +10         -9
  AIR          +11        -10
  AIR          +12        -11
  AIR          +13        -12
  AIR          +14        -13
  END
    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
    1    2    1    2    1    2    1    2    1    2    1    2    1    1    0
22 GROUP AIR  CROSS SECTIONS —— P5 ——   DENSITY = 1.29 G/L
   22   22    0    0   22   25    4    1    1    1    6    3    0
    0    0    0    0    0    0    0    0    0    0    0
 0 +12031- 9 0 +    0+ 0 0 +70247- 9 0 +22067- 921R+    0+ 0 0 +11281- 9        1
 0 +    0+ 0 0 +65243- 9 0 +21194- 9 0 +14055- 920R+    0+ 0 0 +10355- 9        2
 0 +    0+ 0 0 +57910- 9 0 +15527- 9 0 +14377- 9 0 +31171-1019R+    0+ 0        3
 0 +92492-10 0 +    0+ 0 0 +57436- 9 0 +23946- 9 0 +16100- 9 0 +32754-10        4
 0 +29281-1018R+    0+ 0 0 +90745-10 0 +    0+ 0 0 +63100- 9 0 +27823- 9        5
 0 +19310- 9 0 +16113-10 0 +23152-10 0 +34698-1017R+    0+ 0 0 +14621- 9        6
 0 +    0+ 0 0 +72884- 9 0 +29980- 9 0 +22082- 9 0 +70348-11 0 +13142-10        7
 0 +21940-10 0 +25880-1016R+    0+ 0 0 +13259- 9 0 +    0+ 0 0 +86261- 9        8
 0 +42031- 9 0 +27697- 9 0 +28517-10 0 +97520-11 0 +19957-10 0 +33094-10        9
```

Table 3.2 (continued)

```
 0 +32588-1015R+     0+ 0 0 +70810-10 0 +     0+ 0 0 +62171- 9 0 +23374- 9       10
 0 +26782- 9 0 +52500-15 0 +22719-11 0 +56244-11 0 +14450-10 0 +17795-10       11
 0 +17006-1014R+     0+ 0 0 +33771-10 0 +     0+ 0 0 +53538- 9 0 +75580-10       12
 0 +88161-10 0 +31414-10 0 +38131-13 0 +63938-12 0 +12152-11 0 +29969-11       13
 0 +37520-11 0 +35440-1113R+     0+ 0 0 +36572-10 0 +     0+ 0 0 +74575- 9       14
 0 +35996- 9 0 +41220- 9 0 +22817- 9 0 +79882-11 0 +84728-12 0 +28782-11       15
 0 +54518-11 0 +22246-10 0 +16442-10 0 +15359-1012R+     0+ 0 0 +26696-10       16
 0 +     0+ 0 0 +10111- 8 0 +72009- 9 0 +36923- 9 0 +13828-10 0 +     0+ 0       17
 0 +28461-12 0 +25686-11 0 +37451-11 0 +95473-11 0 +20771-10 0 +19773-10       18
 0 +18079-1011R+     0+ 0 0 +13894-10 0 +     0+ 0 0 +10127- 8 0 +80896- 9       19
 0 +26435- 9 3R+     0+ 0 0 +11923-11 0 +16928-11 0 +22410-11 0 +74300-11       20
 0 +71270-11 0 +10911-10 0 +97580-1110R+     0+ 0 0 +23108-11 0 +     0+ 0       21
 0 +15950- 8 0 +14549- 8 0 +18991- 9 3R+     0+ 0 0 +60646-12 0 +97578-12       22
 0 +69178-12 0 +89141-12 0 +31607-11 0 +23569-11 0 +40836-11 0 +35918-11       23
 9R+     0+ 0 0 +56899-12 0 +     0+ 0 0 +27466- 8 0 +26402- 8 0 +13777- 9       24
 4R+     0+ 0 0 +24320-12 0 +50672-13 0 +35016-13 0 +44416-13 0 +10237-12       25
 0 +11424-12 0 +19655-12 0 +17135-12 8R+     0+ 0 0 +27307-11 0 +     0+ 0       26
 0 +35854- 8 0 +33087- 8 0 +10590- 9 5R+     0+ 0 0 +53689-15 0 +61250-16       27
 0 +44470-16 0 +57069-16 0 +99544-16 0 +15180-15 0 +27607-15 0 +24655-15       28
 7R+     0+ 0 0 +70094-11 0 +     0+ 0 0 +38206- 8 0 +35239- 8 0 +27393- 9       29
 6R+     0+ 0 0 +39690-16 0 +38084-17 0 +19369-17 0 +23157-17 0 +28545-17       30
 0 +11206-16 0 +20683-16 0 +16798-16 6R+     0+ 0 0 +14998-10 0 +     0+ 0       31
 0 +39881- 8 0 +35523- 8 0 +26970- 913R+     0+ 0 0 +23985-21 0 +11837-19       32
 5R+     0+ 0 0 +26343-10 0 +     0+ 0 0 +40234- 8 0 +34704- 8 0 +42079- 9       33
14R+     0+ 0 0 +59279-22 0 +24502-20 4R+     0+ 0 0 +46455-10 0 +     0+ 0       34
 0 +40905- 8 0 +36293- 8 0 +52676- 916R+     0+ 0 0 +40545-21 3R+     0+ 0       35
 0 +81039-10 0 +     0+ 0 0 +41602- 8 0 +35628- 8 0 +41478- 920R+     0+ 0       36
 0 +13300- 9 0 +     0+ 0 0 +42106- 8 0 +35798- 8 0 +51629- 920R+     0+ 0       37
 0 +57942- 9 0 +     0+ 0 0 +46612- 8 0 +40818- 8 0 +49769- 920R+     0+ 0       38
 2R+     0+ 0 0 +70247- 9 0 +55366- 923R+     0+ 0 0 +65243- 9 0 +51812- 9        1
 0 +28039-1022R+     0+ 0 0 +57910- 9 0 +46511- 9 0 +82529-12 0 -34105-10        2
21R+     0+ 0 0 +57436- 9 0 +52074- 9 0 -56910-10 0 -33645-1021R+     0+ 0        3
 0 +63100- 9 0 +56259- 9 0 -95744-10 0 -39411-1121R+     0+ 0 0 +72884- 9        4
 0 +67834- 9 0 -16134- 9 0 -86327-1121R+     0+ 0 0 +86261- 9 0 +75179- 9        5
 0 -17914- 9 0 -68136-1021R+     0+ 0 0 +62171- 9 0 +46303- 9 0 -30410- 9        6
22R+     0+ 0 0 +53538- 9 0 +20210- 9 0 +67313-10 0 -77370-1021R+     0+ 0        7
 0 +74575- 9 0 +57669- 9 0 +84142-10 0 -29471- 9 0 -22037-1020R+     0+ 0        8
 0 +10111- 8 0 +64755- 9 0 -25951- 9 0 -38164-1021R+     0+ 0 0 +10127- 8        9
 0 +42570- 9 0 -23334- 922R+     0+ 0 0 +15950- 8 0 +36793- 9 0 -13650- 9       10
22R+     0+ 0 0 +27466- 8 0 +47370- 9 0 -12684- 922R+     0+ 0 0 +35854- 8       11
 0 +76560- 9 0 -95901-1022R+     0+ 0 0 +38206- 8 0 +81388- 9 0 -24833- 9       12
22R+     0+ 0 0 +39881- 8 0 +55671- 9 0 -26278- 922R+     0+ 0 0 +40234- 8       13
 0 +10563- 8 0 -38215- 922R+     0+ 0 0 +40905- 8 0 +96174- 9 0 -17822- 9       14
22R+     0+ 0 0 +41602- 8 0 +10593- 8 0 -37673- 922R+     0+ 0 0 +42106- 8       15
 0 +10270- 8 0 -46872- 922R+     0+ 0 0 +46612- 8 0 +     0+ 0 0 -45480- 9       16
20R+     0+ 0                                                                   17
 2R+     0+ 0 0 +70247- 9 0 +69107- 923R+     0+ 0 0 +65243- 9 0 +62392- 9        1
 0 -72000-1022R+     0+ 0 0 +57910- 9 0 +54463- 9 0 -78847-10 0 +40544-10        2
21R+     0+ 0 0 +57436- 9 0 +56525- 9 0 -69713-10 0 +38568-1021R+     0+ 0        3
 0 +63100- 9 0 +54019- 9 0 -40874-10 0 +56019-1121R+     0+ 0 0 +72884- 9        4
 0 +69678- 9 0 -17789-10 0 +11990-1021R+     0+ 0 0 +86261- 9 0 +74494- 9        5
 0 +15360- 9 0 +76303-1021R+     0+ 0 0 +62171- 9 0 +34577- 9 0 +21974- 9        6
22R+     0+ 0 0 +53538- 9 0 +26501- 9 0 -44031-10 0 +84248-1021R+     0+ 0        7
 0 +74575- 9 0 +28015- 9 0 -15857- 9 0 +43965-10 0 +30911-1020R+     0+ 0        8
 0 +10111- 8 0 +40335- 9 0 -77993-11 0 +53567-1021R+     0+ 0 0 +10127- 8        9
 0 +22675- 9 0 +98275-1022R+     0+ 0 0 +15950- 8 0 +13766- 9 0 -31135-10       10
22R+     0+ 0 0 +27466- 8 0 +48445-10 0 -10203-1022R+     0+ 0 0 +35854- 8       11
 0 +75791-10 0 -90808-1122R+     0+ 0 0 +38206- 8 0 +80502-10 0 -23240-10       12
22R+     0+ 0 0 +39881- 8 0 +93603-10 0 -24438-1022R+     0+ 0 0 +40234- 8       13
 0 +10285- 9 0 -35087-1022R+     0+ 0 0 +40905- 8 0 +93944-10 0 -43949-10       14
```

Table 3.2 (continued)

```
22R+     0+ 0 0 +41602- 8 0 +10258- 9 0 -36279-1022R+     0+ 0 0 +42106- 8       15
 0 +10426- 9 0 -42212-1022R+     0+ 0 0 +46612- 8 0 +     0& 0 0 -34698-10       16
20R+     0+ 0                                                                   17
 2R+     0+ 0 0 +70247- 9 0 +70321- 923R+     0+ 0 0 +65243- 9 0 +62536- 9     1
 0 +13849-1022R+     0+ 0 0 +57910- 9 0 +54199- 9 0 +29572-10 0 -34172-10       2
21R+     0+ 0 0 +57436- 9 0 +54680- 9 0 +50026-10 0 -30256-1021R+     0+ 0      3
 0 +63100- 9 0 +47578- 9 0 +69933-10 0 -61050-1121R+     0+ 0 0 +72884- 9       4
 0 +54800- 9 0 +51915-10 0 -12584-1021R+     0+ 0 0 +86261- 9 0 +54971- 9       5
 0 -76147-10 0 -54466-1021R+     0+ 0 0 +62171- 9 0 +18010- 9 0 -10531- 9       6
22R+     0+ 0 0 +53538- 9 0 +25118- 9 0 -43162-10 0 -55490-1021R+     0+ 0      7
 0 +74575- 9 0 +10308- 9 0 -12354- 9 0 -27625-10 0 -32964-1020R+     0+ 0       8
 0 +10111- 8 0 +19170- 9 0 -44569-10 0 -57015-1021R+     0+ 0 0 +10127- 8       9
 0 +96616-11 0 -57384-1022R+     0+ 0 0 +15950- 8 0 +14942-10 0 -17019-10       10
22R+     0+ 0 0 +27466- 8 0 -29453-10 0 -11580-1122R+     0+ 0 0 +35854- 8       11
 0 -40830-10 0 -22594-1122R+     0+ 0 0 +38206- 8 0 -43758-10 0 -57987-11       12
22R+     0+ 0 0 +39881- 8 0 -43381-10 0 -61287-1122R+     0+ 0 0 +40234- 8       13
 0 -41300-10 0 -88550-1122R+     0+ 0 0 +40905- 8 0 -44037-10 0 -11322-10       14
22R+     0+ 0 0 +41602- 8 0 -41795-10 0 -92138-1122R+     0+ 0 0 +42106- 8       15
 0 -24206-10 0 -11908-1022R+     0+ 0 0 +46612- 8 0 +     0& 0 0 -19669-10       16
20R+     0+ 0                                                                   17
 2R+     0+ 0 0 +70247- 9 0 +63319- 923R+     0+ 0 0 +65243- 9 0 +56598- 9     1
 0 +79906-1022R+     0+ 0 0 +57910- 9 0 +47662- 9 0 +85919-10 0 +23298-10       2
21R+     0+ 0 0 +57436- 9 0 +41757- 9 0 +94751-10 0 +18275-1021R+     0+ 0      3
 0 +63100- 9 0 +31740- 9 0 +50361-10 0 +54739-1121R+     0+ 0 0 +72884- 9       4
 0 +33677- 9 0 -65477-10 0 +10656-1021R+     0+ 0 0 +86261- 9 0 +22392- 9       5
 0 -99578-10 0 +21930-1021R+     0+ 0 0 +62171- 9 0 +99323-10 0 -63973-10       6
22R+     0+ 0 0 +53538- 9 0 +17702- 9 0 -60963-10 0 +15008-1021R+     0+ 0      7
 0 +74575- 9 0 +47284-10 0 -20232- 9 0 +51567-10 0 +28578-1020R+     0+ 0       8
 0 +10111- 8 0 +73559-10 0 -79020-11 0 +49040-1021R+     0+ 0 0 +10127- 8       9
 0 +10166-10 0 -44963-1022R+     0+ 0 0 +15950- 8 0 +93530-11 0 -69195-11       10
22R+     0+ 0 0 +27466- 8 0 +83113-10 0 +22143-1122R+     0+ 0 0 +35854- 8       11
 0 +75510-10 0 +52998-1122R+     0+ 0 0 +38206- 8 0 +80679-10 0 +13709-10       12
22R+     0+ 0 0 +39881- 8 0 +78252-10 0 +14581-1022R+     0+ 0 0 +40234- 8       13
 0 +73370-10 0 +21203-1022R+     0+ 0 0 +40905- 8 0 +79796-10 0 +26730-10       14
22R+     0+ 0 0 +41602- 8 0 +74416-10 0 +21278-1022R+     0+ 0 0 +42106- 8       15
 0 +16455- 9 0 +26766-1022R+     0+ 0 0 +46612- 8 0 +     0& 0 0 +41974-10       16
20R+     0+ 0                                                                   17
 2R+     0+ 0 0 +70247- 9 0 +48470- 923R+     0+ 0 0 +65243- 9 0 +42755- 9     1
 0 -37301-1022R+     0+ 0 0 +57910- 9 0 +32130- 9 0 -55982-10 0 -15807-10       2
21R+     0+ 0 0 +57436- 9 0 +19518- 9 0 -81760-10 0 -11315-1021R+     0+ 0      3
 0 +63100- 9 0 +10832- 9 0 -13409- 9 0 -40247-1121R+     0+ 0 0 +72884- 9       4
 0 +11598- 9 0 -21523-10 0 -71699-1121R+     0+ 0 0 +86261- 9 0 +67571-10       5
 0 -67094-10 0 +78990-1221R+     0+ 0 0 +62171- 9 0 +40592-10 0 -16550-10       6
22R+     0+ 0 0 +53538- 9 0 +78886-10 0 -36437-10 0 +11706-1021R+     0+ 0      7
 0 +74575- 9 0 +61549-11 0 -60885-10 0 +22592-10 0 -19893-1020R+     0+ 0       8
 0 +10111- 8 0 +10090-10 0 -14219-10 0 -33189-1021R+     0+ 0 0 +10127- 8       9
 0 -29733-11 0 -22978-1022R+     0+ 0 0 +15950- 8 0 -10402-10 0 -18486-11       10
22R+     0+ 0 0 +27466- 8 0 -75968-10 0 -20242-1122R+     0+ 0 0 +35854- 8       11
 0 -97342-10 0 -11581-1022R+     0+ 0 0 +38206- 8 0 -10417- 9 0 -30083-10       12
22R+     0+ 0 0 +39881- 8 0 -55899-10 0 -31970-1022R+     0+ 0 0 +40234- 8       13
 0 -85091-10 0 -46541-1022R+     0+ 0 0 +40905- 8 0 -99314-10 0 -58362-10       14
22R+     0+ 0 0 +41602- 8 0 -89912-10 0 -46163-1022R+     0+ 0 0 +42106- 8       15
 0 -46088-10 0 -57320-1022R+     0+ 0 0 +46612- 8 0 +     0& 0 0 -88991-10       16
20R+     0+ 0                                                                   17
    1   -1 1.16
SAMBO ANALYSIS INPUT DATA
     7    0    0    0    0    1    0    1
     0.        0.        1.0E+4
     0.        0.        2.0E+4
     0.        0.        3.0E+4
```

Table 3.2 (continued)

```
      0.          0.          6.0E+4
      0.          0.          7.0E+4
      0.          0.          9.0E+4
      0.          0.         12.0E+4
                        BLANK CARD
4 PI R**2 FLUENCE
1.0         1.0         1.0         1.0         1.0         1.0         1.0         CARDS DD
1.0         1.0         1.0         1.0         1.0         1.0           1.0
1.0         1.0         1.0         1.0         1.0         1.0           1.0
1.0
$$$$$$$$$$    MORSE SAMPLE PROBLEM  ********************
```

```
MORSE SAMPLE PROBLEM  POINT FISSION SOURCE IN AIR
THIS CASE WAS BEGUN ON THURSDAY, AUGUST 22, 1974

   NSTRT    NMOST     NITS    NOUIT   NGPOTN  NGPOTG     NMGP     NMTG  NCOLTP   IADJM  MAXTIM   MEDIA  MEDALB
     200      400       10        1       13       0       22       22       0       0    5.00       1       0

   ISOUR   NGPFS  ISBIAS            WTSTRT       EBOTN       EBOTG        TCUT        VELTH
       0      14       0      1.0000E 00  0.0          0.0          1.0000E 00  2.2000E 05

            XSTRT       YSTRT       ZSTRT      AGSTRT        UINP        VINP        WINP
              0.0         0.0         0.0         0.0         0.0         0.0         0.0

DOF IS DIFFERENT FROM WTSTRT, DOF = 0.98451E 00

                SOURCE DATA
   GROUP     UNNORMALIZED      NORMALIZED
              FRACTION          FRACTION
       1     1.5579E-04        0.000158
       2     8.9338E-04        0.000907
       3     3.4786E-03        0.003533
       4     1.3903E-02        0.014122
       5     3.4557E-02        0.035102
       6     3.5047E-02        0.035599
       7     1.0724E-01        0.108930
       8     8.8963E-02        0.090365
       9     2.3186E-02        0.023551
      10     1.2030E-01        0.122196
      11     2.1803E-01        0.221466
      12     1.9837E-01        0.201496
      13     1.4036E-01        0.142572
      14     1.5489E-02        0.0
   TOTAL     9.9997E-01

GROUP PARAMETERS.  GROUP NUMBERS GREATER THAN   22 CORRESPOND TO SECONDARY PARTICLES

   GROUP      UPPER EDGE          VELOCITY
              (EV)                (CM/SEC)
       1     1.5000E 07        5.1016E 09
       2     1.2214E 07        4.6091E 09
       3     1.0000E 07        4.1705E 09
       4     8.1873E 06        3.7299E 09
       5     6.3600E 06        3.2911E 09
       6     4.9658E 06        2.9389E 09
       7     4.0657E 06        2.6017E 09
       8     3.0119E 06        2.2888E 09
       9     2.4659E 06        2.1461E 09
      10     2.3500E 06        1.9986E 09
      11     1.8268E 06        1.6753E 09
      12     1.1080E 06        1.2593E 09
      13     5.5020E 05        7.9525E 08
      14     1.1109E 05        3.3083E 08
      15     3.3546E 03        6.1365E 07
      16     5.8294E 02        2.5581E 07
      17     1.0130E 02        1.1164E 07
      18     2.9020E 01        6.1615E 06
      19     1.0677E 01        6.2822E 06
      20     3.0790E 01        6.3258E 06
      21     1.1253E 01        3.3403E 06
      22     4.1400E-01        2.2000E 05

INITIAL RANDOM NUMBER = 000035FA731A

NSPLT= 1     NKILL= 1     NPAST= 1     NOLEAK= 0     IESIAS= 0     KXREG= 1     MAXGP= 13

WEIGHT STANDARDS FOR SPLITTING AND RUSSIAN ROULETTE AND PATHLENGTH STRETCHING PARAMETERS

 NGP1  NDG NGP2  NRG1 NDRG NRG2     WTHIM1       WTLOW1       WTAVE1       XNU
    0    0    0     0    0    0  1.0000E 01  1.0000E-02  1.0000E-01  0.5000E 00

NSOUR= 0     MFISTP= 0     NKCALC= 0     NORMF= 0
```

SAMPLE PROB. 1 FOR MORSE

IVOPT = 0 IDBG = 0

BODY DATA

SPH	1	0.0	0.0	0.0	0.33000000D 04	0.0	0.0	3
SPH	2	0.0	0.0	0.0	0.50000000D 04	0.0	0.0	11
SPH	3	0.0	0.0	0.0	0.75000000D 04	0.0	0.0	19
SPH	4	0.0	0.0	0.0	0.10000000D 05	0.0	0.0	27
SPH	5	0.0	0.0	0.0	0.15000000D 05	0.0	0.0	35
SPH	6	0.0	0.0	0.0	0.20000000D 05	0.0	0.0	43
SPH	7	0.0	0.0	0.0	0.30000000D 05	0.0	0.0	51
SPH	8	0.0	0.0	0.0	0.60000000D 05	0.0	0.0	59
SPH	9	0.0	0.0	0.0	0.70000000D 05	0.0	0.0	67
SPH	10	0.0	0.0	0.0	0.90000000D 05	0.0	0.0	75
SPH	11	0.0	0.0	0.0	0.12000000D 06	0.0	0.0	83
SPH	12	0.0	0.0	0.0	0.15000000D 06	0.0	0.0	91
SPH	13	0.0	0.0	0.0	0.10000000D 07	0.0	0.0	99
SPH	14	0.0	0.0	0.0	0.10000000D 08	0.0	0.0	107
END	15	0.0	0.0	0.0	0.0	0.0	0.0	115

NUMBER OF BODIES 14
LENGTH OF FPD-ARRAY 120

INPUT ZONE DATA

AIR	0	1	0	0	0	0	0	0	0	Z	1
AIR	0	2	-1	0	0	0	0	0	0	Z	2
AIR	0	3	-2	0	0	0	0	0	0	Z	3
AIR	0	4	-3	0	0	0	0	0	0	Z	4
AIR	0	5	-4	0	0	0	0	0	0	Z	5
AIR	0	6	-5	0	0	0	0	0	0	Z	6
AIR	0	7	-6	0	0	0	0	0	0	Z	7
AIR	0	8	-7	0	0	0	0	0	0	Z	8
AIR	0	9	-8	0	0	0	0	0	0	Z	9
AIR	0	10	-9	0	0	0	0	0	0	Z	10
AIR	0	11	-10	0	0	0	0	0	0	Z	11
AIR	0	12	-11	0	0	0	0	0	0	Z	12
AIR	0	13	-12	0	0	0	0	0	0	Z	13
AIR	0	14	-13	0	0	0	0	0	0	Z	14
END	0	0	0	0	0	0	0	0	0	Z	15

NUMBER OF INPUT ZONES 14
NUMBER OF CODE ZONES 14
LENGTH OF INTEGER ARRAY 361

CODE ZONE	INPUT ZONE	ZONE DATA LOC.	NO. OF BODIES	REGION NO.	MEDIA NO.
1	1	99	1	1	1
2	2	104	2	1	2
3	3	113	2	1	1
4	4	122	2	1	2
5	5	131	2	1	1
6	6	140	2	1	2
7	7	149	2	1	1
8	8	158	2	1	2
9	9	167	2	1	1
10	10	176	2	1	2
11	11	185	2	1	1
12	12	194	2	1	2
13	13	203	2	1	1
14	14	212	2	1	0

I	KR1(I)	KR2(I)
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14

MORSE REGION IN INPUT ZONE(I) ARRAY MRIZ(I),I=1,14)

1 1 1 1 1 1 1 1 1 1 1 1 1 1

MORSE MEDIA IN INPUT ZONE(I) ARRAY MMIZ(I),I=1,14)

1 2 1 2 1 2 1 2 1 2 1 2 1 0

OPTION 0 WAS USED IN CALCULATING VOLUMES. FOR 1 REGIONS
0-SET VOLUMES = 1. 1-CONCENTRIC SPHERES. 2-SLABS. 3-INPUTVOLUMES.

VOLUMES (CM**) USED IN COLLISIONS DENSITY AND TRACK LENGTH ESTIMATORS.

REG	1
VOLUME	1.000D 00

NGEOM= 551. NGLAST= 1294

22 GROUP AIR CROSS SECTIONS --- P5 --- DENSITY = 1.29 G/L

NUMBER OF PRIMARY GROUPS (NGP)	22
NUMBER OF PRIMARY DOWNSCATTERS (NDS)	22
NUMBER OF SECONDARY GROUPS (NGG)	0
NUMBER OF SECONDARY DOWNSCATTERS (NDSG)	0
NUMBER OF PRIM+SEC GROUPS (INGP)	22
TABLE LENGTH (ITBL)	25
LOC OF WITHIN GROUP (SIG GG) (ISGG)	4
NUMBER OF MEDIA (NMED)	1
NUMBER OF INPUT ELEMENTS (NELEM)	1
NUMBER OF MIXING ENTRIES (NMIX)	1
NUMBER OF COEFFICIENTS (NCOEF)	6
NUMBER OF ANGLES (NSCT)	3
RESTORE COEFF (ISTAT)	0
ADJOINT SWITCH (FROM MORSE)	0

INPUT/OUTPUT OPTIONS

IRDSG (AS READ)	0
ISTR (AS STORE)	0
IFMU (MUS)	0
IMOM (MOMENTS)	0
IPRIN (ANGLES,PROB)	0
IPUN (IMPOSSIBLE COEF)	0
CARD FORMAT (IDTF)	0
INPUT TAPE (IXTAPE)	0
MORSEC TAPE (JXTAPE)	0
06R TAPE (I06RT)	0

STORAGE ALLOCATIONS

CROSS SECTIONS START AT	1295	
LAST LOCATION USED (PERM)	3245	
TEMP LOCATIONS USED	13207 TO	15000
EXCESS STORAGE (TEMP)	9962	

MIXING TABLE

MEDIA 1 CONTAINS ELEMENT 1 WITH DENSITY 1.1600E 00

CROSS SECTIONS FOR MEDIA 1

GROUP	SIGT	SIGS1	PNUP	PNABS	GAMGEN	NU*FIS	DOWNSCATTER PROBABILITY
1	8.149E-05	6.753E-05	0.0	0.8287	0.0	0.0	0.3791 0.2414 0.0535 0.0503 0.0593 0.0445 0.0560 0.0292 0.0061 0.0264 0.0311 0.0168 0.0062 0.0003 0.0000 0.0000 0.0000 0.0000 0.0000 0.0 0.0 0.0
2	7.568E-05	6.259E-05	0.0	0.8271	0.0	0.0	0.3928 0.2664 0.0607 0.0429 0.0407 0.0613 0.0330 0.0070 0.0305 0.0366 0.0202 0.0076 0.0004 0.0000 0.0000 0.0000 0.0000 0.0 0.0 0.0
3	6.718E-05	5.516E-05	0.0	0.8212	0.0	0.0	0.4106 0.3386 0.0339 0.0276 0.0420 0.0304 0.0063 0.0468 0.0437 0.0150 0.0050 0.0002 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0
4	6.663E-05	5.590E-05	0.0	0.8390	0.0	0.0	0.4969 0.4007 0.0146 0.0202 0.0117 0.0025 0.0113 0.0198 0.0154 0.0066 0.0002 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0
5	7.320E-05	6.267E-05	0.0	0.8562	0.0	0.0	0.5150 0.4087 0.0528 0.0042 0.0012 0.0053 0.0069 0.0041 0.0016 0.0001 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0
6	8.455E-05	6.759E-05	0.0	0.7994	0.0	0.0	0.5146 0.4754 0.0000 0.0001 0.0015 0.0044 0.0029 0.0012 0.0001 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0
7	1.001E-04	8.468E-05	0.0	0.8463	0.0	0.0	0.5757 0.3669 0.0430 0.0109 0.0004 0.0016 0.0013 0.0001 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0 0.0
8	7.212E-05	6.391E-05	0.0	0.8861	0.0	0.0	0.4243 0.1600 0.4142 0.0 0.0 0.0011 0.0004 0.0000 0.0000 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
9	6.210E-05	5.819E-05	0.0	0.9369	0.0	0.0	0.1507 0.8218 0.0276 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
10	8.651E-05	8.227E-05	0.0	0.9510	0.0	0.0	0.5076 0.4924 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
11	1.173E-04	1.142E-04	0.0	0.9736	0.0	0.0	0.7315 0.2685 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
12	1.175E-04	1.159E-04	0.0	0.9863	0.0	0.0	0.8099 0.1901 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
13	1.850E-04	1.847E-04	0.0	0.9985	0.0	0.0	0.9135 0.0865 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
14	3.186E-04	3.185E-04	0.0	0.9998	0.0	0.0	0.9614 0.0386 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
15	4.159E-04	4.156E-04	0.0	0.9992	0.0	0.0	0.9235 0.0765 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
16	4.432E-04	4.424E-04	0.0	0.9982	0.0	0.0	0.9240 0.0760 0.0 0.0 0.0 0.0 0.0
17	4.626E-04	4.609E-04	0.0	0.9962	0.0	0.0	0.8941 0.1059 0.0 0.0 0.0 0.0
18	4.667E-04	4.637E-04	0.0	0.9935	0.0	0.0	0.8682 0.1318 0.0 0.0 0.0
19	4.745E-04	4.691E-04	0.0	0.9887	0.0	0.0	0.8974 0.1026 0.0 0.0
20	4.826E-04	4.732E-04	0.0	0.9805	0.0	0.0	0.8734 0.1266 0.0
21	4.884E-04	4.730E-04	0.0	0.9684	0.0	0.0	0.8779 0.1221
22	5.407E-04	4.735E-04	0.0	0.8757	0.0	0.0	1.0000

BANKS START AT 3246
LAST LOCATION USED 8045

SAMBO ANALYSIS INPUT DATA

ND= 7. NNE= 0. NE= 0. NT= 0. NA= 0. NRESP= 1. NEX= 0. NEXND= 1

DET	X	Y	Z	RAD	TO
1	0.0	0.0	1.0000E 04	1.0000E 04	1.9602E-06
2	0.0	0.0	2.0000E 04	2.0000E 04	3.9204E-06
3	0.0	0.0	3.0000E 04	3.0000E 04	5.8805E-06
4	0.0	0.0	6.0000E 04	6.0000E 04	1.1761E-05
5	0.0	0.0	7.0000E 04	7.0000E 04	1.3721E-05
6	0.0	0.0	9.0000E 04	9.0000E 04	1.7642E-05
7	0.0	0.0	1.2000E 05	1.2000E 05	2.3522E-05

GROUP	RESP(1)
1	1.0000E 00
2	1.0000E 00
3	1.0000E 00
4	1.0000E 00
5	1.0000E 00
6	1.0000E 00
7	1.0000E 00
8	1.0000E 00
9	1.0000E 00
10	1.0000E 00
11	1.0000E 00
12	1.0000E 00
13	1.0000E 00
14	1.0000E 00
15	1.0000E 00
16	1.0000E 00
17	1.0000E 00
18	1.0000E 00
19	1.0000E 00
20	1.0000E 00
21	1.0000E 00
22	1.0000E 00

NUMBER OF PRIMARY ENERGY BINS 0
TOTAL NUMBER OF ENERGY BINS 0

NUMBER OF TIME BINS 0

NUMBER OF ANGLE BINS 0
UPPER LIMITS OF COSINE BINS

233 CELLS USED BY ANALYSIS. 6722 CELLS REMAIN UNUSED.

```
TIME REQUIRED FOR INPUT WAS 1 SECOND.
YOU ARE USING THE DEFAULT VERSION OF STRUN WHICH DOES NOTHING.

***START BATCH   1                          RANDOM=000035FA731A

SOURCE DATA
YOU ARE USING THE DEFAULT VERSION OF SOURCE WHICH SETS WATE TO 00F AND PROVIDES AN ENERGY EG.
  WTAVE        IAVE     UAVE      VAVE      WAVE       XAVE       YAVE       ZAVE       AGEAVE
 1.969E 02    10.38  -0.0176  -0.0082   0.0450    0.0        0.0        0.0        0.0

NUMBER OF COLLISIONS OF TYPE NCOLL
  SOURCE SPLIT(D)    FISHN    GAMGEN REALCOLL    ALBEDO    BDRYX    ESCAPE    E-CUT TIMEKILL R R KILL R R SURV  GAMLOST
      200            4          0         0      3810         0      2174         0      179         0       25         1         0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH   2                          RANDOM=E84E9DDF806A

SOURCE DATA
  WTAVE        IAVE     UAVE      VAVE      WAVE       XAVE       YAVE       ZAVE       AGEAVE
 1.969E 02    10.12  -0.0412  -0.0616   0.0136    0.0        0.0        0.0        0.0

NUMBER OF COLLISIONS OF TYPE NCOLL
  SOURCE SPLIT(D)    FISHN    GAMGEN REALCOLL    ALBEDO    BDRYX    ESCAPE    E-CUT TIMEKILL R R KILL R R SURV  GAMLOST
      200            0          0         0      3751         0      2107         0      175         0       25         1         0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH   3                          RANDOM=59B09313CEC2

SOURCE DATA
  WTAVE        IAVE     UAVE      VAVE      WAVE       XAVE       YAVE       ZAVE       AGEAVE
 1.969E 02    10.08  -0.0285  -0.0429  -0.0055    0.0        0.0        0.0        0.0

NUMBER OF COLLISIONS OF TYPE NCOLL
  SOURCE SPLIT(D)    FISHN    GAMGEN REALCOLL    ALBEDO    BDRYX    ESCAPE    E-CUT TIMEKILL R R KILL R R SURV  GAMLOST
      200            6          0         0      3906         0      2161         0      184         0       22         5         0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH   4                          RANDOM=D5B303FBD12A

SOURCE DATA
  WTAVE        IAVE     UAVE      VAVE      WAVE       XAVE       YAVE       ZAVE       AGEAVE
 1.969E 02    10.12    0.0133    0.0904  -0.0035    0.0        0.0        0.0        0.0

NUMBER OF COLLISIONS OF TYPE NCOLL
  SOURCE SPLIT(D)    FISHN    GAMGEN REALCOLL    ALBEDO    BDRYX    ESCAPE    E-CUT TIMEKILL R R KILL R R SURV  GAMLOST
      200            3          0         0      3835         0      2162         0      181         0       22         1         0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH   5                          RANDOM=7D0ED84F208A

SOURCE DATA
  WTAVE        IAVE     UAVE      VAVE      WAVE       XAVE       YAVE       ZAVE       AGEAVE
 1.969E 02    10.64    0.0264    0.0154  -0.0155    0.0        0.0        0.0        0.0

NUMBER OF COLLISIONS OF TYPE NCOLL
  SOURCE SPLIT(D)    FISHN    GAMGEN REALCOLL    ALBEDO    BDRYX    ESCAPE    E-CUT TIMEKILL R R KILL R R SURV  GAMLOST
      200            2          0         0      3509         0      2059         0      191         0       11         1         0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.
```

***START BATCH 6 RANDOM=04A069A9AF0A

SOURCE DATA

WTAVE	IAVE	UAVE	VAVE	WAVE	XAVE	YAVE	ZAVE	AGEAVE
1.969E 02	10.09	0.0121	-0.0250	-0.0234	0.0	0.0	0.0	0.0

NUMBER OF COLLISIONS OF TYPE NCOLL

SOURCE SPLIT(D)	FISHN	GAMGEN	REALCOLL	ALBEDO	BDRYX	ESCAPE	E-CUT	TIMEKILL	R R KILL	R R SURV	GAMLOST
200	2	0	0	3926	0	2192	0	189	0	13	1	0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH 7 RANDOM=C3A5A4BDA252

SOURCE DATA

WTAVE	IAVE	UAVE	VAVE	WAVE	XAVE	YAVE	ZAVE	AGEAVE
1.969E 02	9.99	0.0411	0.0326	0.0450	0.0	0.0	0.0	0.0

NUMBER OF COLLISIONS OF TYPE NCOLL

SOURCE SPLIT(D)	FISHN	GAMGEN	REALCOLL	ALBEDO	BDRYX	ESCAPE	E-CUT	TIMEKILL	R R KILL	R R SURV	GAMLOST
200	11	0	0	3931	0	2219	0	173	0	38	1	0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH 8 RANDOM=21360945E80A

SOURCE DATA

WTAVE	IAVE	UAVE	VAVE	WAVE	XAVE	YAVE	ZAVE	AGEAVE
1.969E 02	10.22	0.0145	-0.0039	0.0701	0.0	0.0	0.0	0.0

NUMBER OF COLLISIONS OF TYPE NCOLL

SOURCE SPLIT(D)	FISHN	GAMGEN	REALCOLL	ALBEDO	BDRYX	ESCAPE	E-CUT	TIMEKILL	R R KILL	R R SURV	GAMLOST
200	6	0	0	3785	0	2150	0	184	0	22	2	0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH 9 RANDOM=8445E9E56E22

SOURCE DATA

WTAVE	IAVE	UAVE	VAVE	WAVE	XAVE	YAVE	ZAVE	AGEAVE
1.969E 02	10.08	0.0219	0.0525	-0.0242	0.0	0.0	0.0	0.0

NUMBER OF COLLISIONS OF TYPE NCOLL

SOURCE SPLIT(D)	FISHN	GAMGEN	REALCOLL	ALBEDO	BDRYX	ESCAPE	E-CUT	TIMEKILL	R R KILL	R R SURV	GAMLOST
200	0	0	0	3719	0	2101	0	180	0	20	2	0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

***START BATCH 10 RANDOM=282172044EE2

SOURCE DATA

WTAVE	IAVE	UAVE	VAVE	WAVE	XAVE	YAVE	ZAVE	AGEAVE
1.969E 02	9.87	0.0081	-0.0068	-0.0391	0.0	0.0	0.0	0.0

NUMBER OF COLLISIONS OF TYPE NCOLL

SOURCE SPLIT(D)	FISHN	GAMGEN	REALCOLL	ALBEDO	BDRYX	ESCAPE	E-CUT	TIMEKILL	R R KILL	R R SURV	GAMLOST
200	9	0	0	3960	0	2345	0	185	0	24	0	0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 3 SECONDS.

THIS CASE WAS RUN ON THURSDAY, AUGUST 22, 1974

4 PI R**2 FLUENCE

RESPONSES(DETECTOR)

DETECTOR	UNCOLL RESPONSE	FSD UNCOLL	TOTAL RESPONSE	FSD TOTAL
1	3.3580E-01	0.00742	2.0546E 00	0.15988
2	1.2503E-01	0.01327	2.1038E 00	0.09497
3	4.9583E-02	0.01819	1.7256E 00	0.07938
4	3.9769E-03	0.02940	6.0785E-01	0.07421
5	1.8191E-03	0.03217	4.5291E-01	0.08572
6	4.0342E-04	0.03656	1.4730E-01	0.09147
7	4.6489E-05	0.04113	4.2184E-02	0.21754

EXTRA ARRAYS OF LENGTH ND

EXT1(7)							
(1)	2447	2632	2467	1713	1488	958	436

TIME REQUIRED FOR THE PRECEDING 10 BATCHES WAS 36 SECONDS.

NEUTRON DEATHS	NUMBER	WEIGHT
KILLED BY RUSSIAN ROULETTE	222	0.14958E 01
ESCAPED	0	0.0
REACHED ENERGY CUTOFF	1821	0.15379E 04
REACHED TIME CUTOFF	0	0.0

NUMBER OF SCATTERINGS

MEDIUM	NUMBER
1	38132
TOTAL	38132

REAL SCATTERING COUNTERS

ENERGY GROUP	REGION 1 NUMBER	WEIGHT
1	0	0.0
2	5	1.26E 00
3	12	6.87E 00
4	57	5.06E 01
5	171	1.19E 02
6	286	1.87E 02
7	879	5.63E 02
8	819	5.29E 02
9	258	1.68E 02
10	1627	1.09E 03
11	4594	3.28E 03
12	8325	6.34E 03
13	21099	1.85E 04
14	0	0.0
15	0	0.0
16	0	0.0
17	0	0.0
18	0	0.0
19	0	0.0
20	0	0.0
21	0	0.0
22	0	0.0

NUMBER OF SPLITTINGS

ENERGY GROUP	REGION 1 NUMBER	WEIGHT
1	0	0.0
2	0	0.0
3	0	0.0
4	0	0.0
5	0	0.0
6	0	0.0
7	0	0.0
8	0	0.0
9	0	0.0
10	0	0.0
11	0	0.0
12	2	1.40E 01
13	41	2.47E 02

NUMBER OF SPLITTINGS PREVENTED BY LACK OF ROOM

ENERGY GROUP	REGION 1 NUMBER	WEIGHT
1	0	0.0
2	0	0.0
3	0	0.0
4	0	0.0
5	0	0.0
6	0	0.0
7	0	0.0
8	0	0.0
9	0	0.0
10	0	0.0
11	0	0.0
12	0	0.0
13	0	0.0

NUMBER OF RUSSIAN ROULETTE KILLS

ENERGY GROUP	REGION 1 NUMBER	WEIGHT
1	0	0.0
2	0	0.0
3	0	0.0
4	0	0.0
5	1	4.16E-03
6	0	0.0
7	6	3.42E-02
8	7	4.03E-02
9	4	2.51E-02
10	6	4.05E-02
11	24	1.66E-01
12	50	3.42E-01
13	124	8.44E-01

NUMBER OF RUSSIAN ROULETTE SURVIVALS

ENERGY GROUP	REGION 1 NUMBER	WEIGHT
1	0	0.0
2	0	0.0
3	0	0.0
4	0	0.0
5	0	0.0
6	0	0.0
7	1	8.74E-03
8	0	0.0
9	0	0.0
10	0	0.0
11	3	2.34E-02
12	2	1.15E-02
13	9	7.15E-02

** NEXT RANDOM NUMBER IS 118541A2E48A

TOTAL CPU TIME FOR THIS PROBLEM WAS 0.61 MINUTES. THE REGION USED WAS 256K.

$$$$$$$$$$ MORSE SAMPLE PROBLEM **********

3.3 PICTURE SAMPLE PROBLEM

This problem illustrates the use of the PICTURE[1] program[†] for looking at the geometry of a given problem. The geometry being illustrated is a tank. This tank model was constructed purely as an illustration of the combinatorial geometry and is in no way accurate or detailed. Both the use of the ØR operator and the ARB body is demonstrated. The input data is listed in Table 3.3. The picture produced by the problem follows.

[†]See description of PICTURE in Part 5 of this document.

Table 3.3. Listing of Input Cards for PICTURE Sample Problem

```
   7     1
.1ETA0
   0     0          COMBINATORIAL GEOMETRY TANK SAMPLE PROBLEM
 ARB       1.25      -1.25      .5        1.25      -2.0       0.0
           1.25       2.0       0.0       1.25       1.25      .5
           -1.25     -1.25      .5        -1.25     -2.0        0.0
           -1.25      2.0       0.0       -1.25      1.25      .5
           1234.      4158.     6587.     2673.      5621.     4378.
 ELL       0.0       -.5        .5        0.0        .5        .5
           1.5
 TRC       0.0        0.0       .8        0.0        2.5       0.0
           .1         .05
 ARB       1.25      -2.0       0.0       1.25      -1.8       -.8
           1.25       2.0       0.0       1.25       1.8       -.8
          -1.25       2.0       0.0      -1.25       1.8       -.8
          -1.25      -2.0       0.0      -1.25      -1.8       -.8
           1243.      7135.     8756.     2864.      7821.     3465.
 BOX       1.25      -1.979     -.6895    0.0        .358      .179
           0.0        -.2       .4        -2.5       0.0       0.0
 BOX       1.25       1.979     -.6895    0.0        -.358     .179
           0.0        .2        .4        -2.5       0.0       0.0
 RCC       1.25      -2.0       -.2       -2.5       0.0       0.0
           .2
 RCC       1.25      -1.8       -.6       -2.5       0.0       0.0
           .2
 RCC       1.25       2.0       -.2       -2.5       0.0       0.0
           .2
 RCC       1.25       1.80      -.6       -2.5       0.0       0.0
           .2
 RCC       1.25      -.9        -.6       -2.5       0.0       0.0
           .2
 RCC       1.25      -.45       -.2       -2.5       0.0       0.0
           .2
 RCC       1.25       0.0       -.6       -2.5       0.0       0.0
           .2
 RCC       1.25       .45       -.2       -2.5       0.0       0.0
           .2
 RCC       1.25       .9        -.6       -2.5       0.0       0.0
           .2
 RPP       -10.       10.       -10.      10.        -10.      10.
 END
   1           +1
   2           +2      -1
   3           +3      -2
   4     OR    +4      -7      -8      -9     -10     -11     -12     -13     -14
             -150R     +5      -7      -8OR   +6      -9      -10
   5     OR    +7OR    +8OR    +9OR   +100R  +11OR   +12CR  +13CR  +140R   +15
   6          +16      -1      -2      -3     -4      -5      -6      -7      -8
              -9      -10
 END
   1    1     1     1     1     1
   1    2     3     4     5  1000
0 0   THIS IS A COMBINATORIAL GEOMETRY TANK.
0.0         -3.0        -2.0        0.0         3.0         2.0
0.0          1.0         0.0        0.0         0.0         1.0
      130
```

COMBINATORIAL GEOMETRY TANK SAMPLE PROBLEM

IVOPT = 0 IDBG = 0

BODY DATA

```
ARB     1  0.12500000 01 -0.12500000 01  0.50000000 00  0.12500000 01 -0.20000000 01  0.0                3
           0.12500000 01  0.20000000 01  0.0            0.12500000 01  0.12500000 01  0.50000000 00
          -0.12500000 01 -0.12500000 01  0.50000000 00 -0.12500000 01 -0.20000000 01  0.0
          -0.12500000 01  0.20000000 01  0.0           -0.12500000 01  0.12500000 01  0.50000000 00
           0.12340000 04  0.41580000 04  0.65870000 04  0.26730000 04  0.56210000 04  0.43760000 04
                         -0.10000000 01  0.0            0.0            0.12500000 01
                          0.0            0.0           -0.10000000 01  0.50000000 00
                          0.10000000 01  0.0            0.0            0.12500000 01
                          0.0            0.0           0.10000000 01  0.0
                          0.0            0.55470020 00 -0.83235030 00  0.11094000 01
                          0.0           -0.55470020 00 -0.83235030 00  0.11094000 01
                          0.90138780 00  0.60000000 01
ELL     2  0.0           -0.50000000 00  0.50000000 00  0.0            0.50000000 00  0.50000000 00   35
           0.15000000 01
TRC     3  0.0            0.0            0.80000000 00  0.0            0.25000000 01  0.0              44
           0.10000000 00  0.50000000-01
ARB     4  0.12500000 01 -0.20000000 01  0.0            0.12500000 01 -0.18000000 01 -0.80000000 00   54
           0.12500000 01  0.20000000 01  0.0            0.12500000 01  0.18000000 01 -0.80000000 00
          -0.12500000 01  0.20000000 01  0.0           -0.12500000 01  0.18000000 01 -0.80000000 00
          -0.12500000 01 -0.20000000 01  0.0           -0.12500000 01 -0.18000000 01 -0.80000000 00
           0.12430000 04  0.71350000 04  0.87560000 04  0.28640000 04  0.78270000 04  0.34650000 04
                         -0.10000000 01  0.0            0.0            0.12500000 01
                          0.0            0.0           -0.10000000 01  0.0
                          0.10000000 01  0.0            0.0            0.12500000 01
                          0.0            0.0           0.10000000 01  0.80000000 00
                          0.0            0.97014250 00  0.24253560 00  0.19402850 01
                          0.0           -0.97014250 00  0.24253560 00  0.19402850 01
                          0.82462110 00  0.60000000 01
BOX     5  0.12500000 01 -0.19790000 01 -0.68950000 00  0.0            0.35800000 00  0.17900000 00   86
           0.0           -0.20000000 00  0.40000000 00 -0.25000000 01  0.0            0.0
BOX     6  0.12500000 01  0.19790000 01 -0.68950000 00  0.0           -0.25800000 00  0.17900000 00  100
           0.0            0.20000000 00  0.40000000 00 -0.25000000 01  0.0            0.0
RCC     7  0.12500000 01 -0.20000000 01 -0.20000000 00 -0.25000000 01  0.0            0.0             114
           0.20000000 00
RCC     8  0.12500000 01 -0.18000000 01 -0.60000000 00 -0.25000000 01  0.0            0.0             123
           0.20000000 00
RCC     9  0.12500000 01  0.20000000 01 -0.20000000 00 -0.25000000 01  0.0            0.0             132
           0.20000000 00
RCC    10  0.12500000 01  0.18000000 01 -0.60000000 00 -0.25000000 01  0.0            0.0             141
           0.20000000 00
RCC    11  0.12500000 01 -0.90000000 00 -0.60000000 00 -0.25000000 01  0.0            0.0             150
           0.20000000 00
RCC    12  0.12500000 01 -0.45000000 00 -0.20000000 00 -0.25000000 01  0.0            0.0             159
           0.20000000 00
RCC    13  0.12500000 01  0.0           -0.60000000 00 -0.25000000 01  0.0            0.0             168
           0.20000000 00
RCC    14  0.12500000 01  0.45000000 00 -0.20000000 00 -0.25000000 01  0.0            0.0             177
           0.20000000 00
RCC    15  0.12500000 01  0.90000000 00 -0.60000000 00 -0.25000000 01  0.0            0.0             186
           0.20000000 00
RPP    16 -0.10000000 02  0.10000000 02 -0.10000000 02  0.10000000 02 -0.10000000 02  0.10000000 02  195
END    17  0.0            0.0            0.0            0.0            0.0            0.0             203
NUMBER OF BODIES     16
LENGTH OF FPD-ARRAY  208
```

INPUT ZONE DATA

1	0	1	0	0	0	0	0	0	0	0	Z	1
2	0	2	-1	0	0	0	0	0	0	0	Z	2
3	0	3	-2	0	0	0	0	0	0	0	Z	3
4	00R	4	-7	-8	-9	-10	-11	-12	-13	-14	Z	4
	0	-15OR	5	-7	-8OR	6	-9	-10	0	0	Z	5
5	00R	7OR	8OR	9OR	10OR	11OR	12OR	13OR	14OR	15	Z	7
6	0	16	-1	-2	-3	-4	-5	-6	-7	-8	Z	16
	0	-9	-10	0	0	0	0	0	0	0	Z	17
END	0	0	0	0	0	0	0	0	0	0	Z	17

NUMBER OF INPUT ZONES 6
NUMBER OF CODE ZONES 16
LENGTH OF INTEGER ARRAY 353

CODE ZONE	INPUT ZONE	ZONE DATA LOC.	NO. OF BODIES	REGION NO.	MEDIA NO.
1	1	113	1	1	1
2	2	118	2	1	2
3	3	127	2	1	3
4	4	136	10	1	4
5	4	177	3	1	4
6	4	190	3	1	4
7	5	203	1	1	5
8	5	208	1	1	5
9	5	213	1	1	5
10	5	218	1	1	5
11	5	223	1	1	5
12	5	228	1	1	5
13	5	233	1	1	5
14	5	238	1	1	5
15	5	243	1	1	5
16	6	248	11	1	1000

I	KR1(I)	KR2(I)
1	1	1
2	2	2
3	3	3
4	4	6
5	7	15
6	16	16

MORSE REGION IN INPUT ZONE(I) ARRAY MRIZ(I),I=1, 6)

1 1 1 1 1 1

MORSE MEDIA IN INPUT ZONE(I) ARRAY MMIZ(I),I=1, 6)

1 2 3 4 5100C

OPTION 0 WAS USED IN CALCULATING VOLUMES, FOR 1 REGIONS
0-SET VOLUMES = 1, 1-CONCENTRIC SPHERES, 2-SLABS, 3-INPUTVOLUMES.

VOLUMES (CM**) USED IN COLLISIONS DENSITY AND TRACK LENGTH ESTIMATORS.

REG	1
VOLUME	1.0000 00

THIS IS A COMBINATORIAL GEOMETRY TANK.

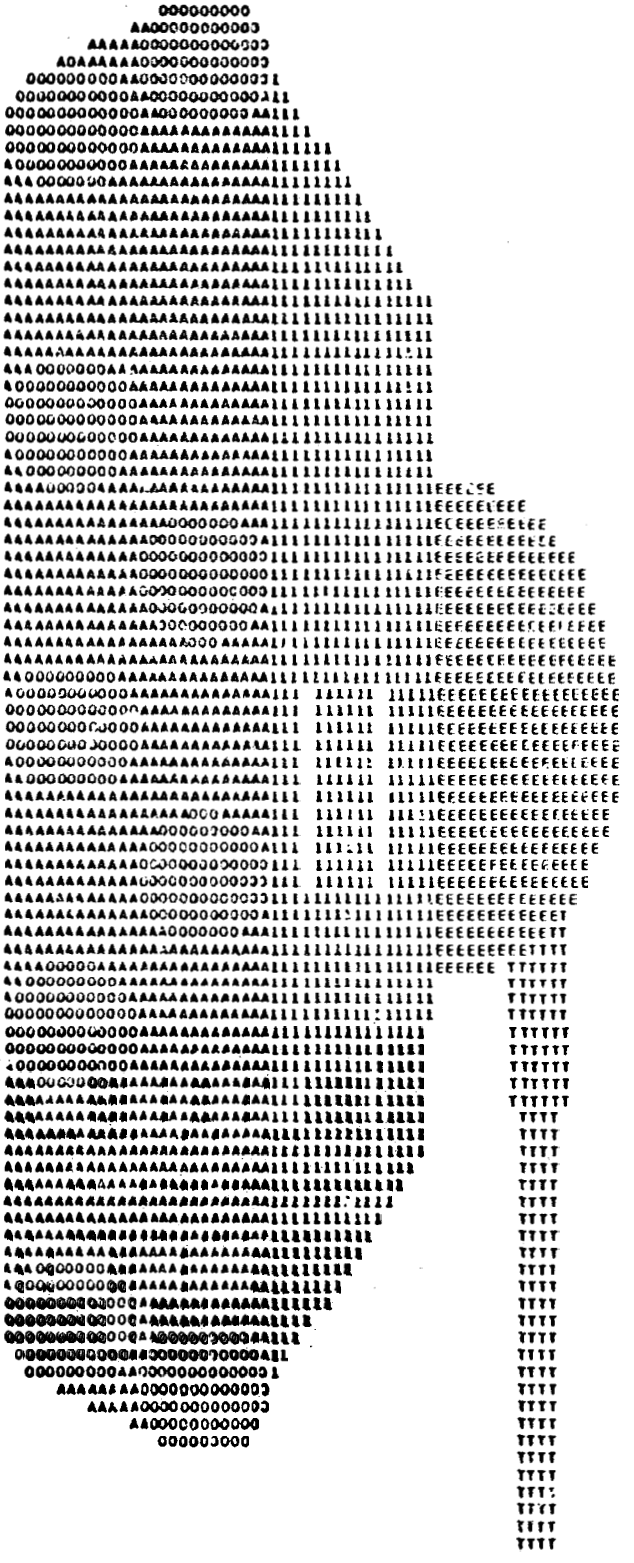
ZONE GEOMETRY

THE SELECTED ATABLE VALUES ARE
. I E T A 0

	UPPER LEFT COORDINATES	LOWER RIGHT COORDINATES
X	0.0	0.0
Y	-0.3000E 01	0.3000E 01
Z	-0.2000E 01	0.2000E 01

	U AXIS %DOWN<	V AXIS %ACROSS<
X	0.0	0.0
Y	1.00000	0.0
Z	0.0	1.00000

NU# 118 NV# 130 DELU# 0.5128E-01 DELV# 0.3077E-01



3.4 REFERENCES

1. D. C. Irving and G. W. Morrison, "PICTURE: An Aid in Debugging Geometry Input Data," ORNL-TM-2892 (May 1970).

PART IV

MORSE

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

4.1. ABSTRACT

The MORSE code is a multipurpose neutron and gamma-ray transport Monte Carlo code. Through the use of multigroup cross sections, the solution of neutron, gamma-ray, or coupled neutron-gamma-ray problems may be obtained in either the forward or adjoint mode. Time dependence for both shielding and criticality problems is provided. General three-dimensional geometry may be used with an albedo option available at any material surface.

Standard multigroup cross sections such as those used in discrete ordinates codes may be used as input; either ANISN or DTF-IV cross-section formats are acceptable. Anisotropic scattering is treated for each group-to-group transfer by utilizing a generalized Gaussian quadrature technique. The modular form of the code with built-in analysis capability for all types of estimators makes it possible to solve a complete neutron-gamma-ray problem as one job and without the use of tapes.

A detailed discussion of the relationship between forward and adjoint flux and collision densities, as well as a detailed description of the treatment of the angle of scattering, is given.

4.2. INTRODUCTION

The Multigroup Oak Ridge Stochastic Experiment code (MORSE) is a multipurpose neutron and gamma-ray transport Monte Carlo code. Some of its features include the ability to treat the transport of either neutrons or gamma rays or a coupled neutron and secondary gamma-ray problem, the incorporation of multigroup cross sections, an option of solving either the forward or adjoint problem, modular input-output, cross section, analysis and geometry modules, debugging routines, time dependence for both shielding and criticality problems, albedo option at any material boundary, three-dimensional combinatorial geometry package, and several types of optional importance sampling.

Traditionally, Monte Carlo codes for solving neutron and gamma-ray transport problems have been separate codes. This has been due to the physics of the interaction processes and the corresponding cross-section information required. However, when multigroup cross sections are employed, the energy group to energy group transfers contain the cross sections for all processes. Also, for anisotropic scattering each group-to-group transfer has an associated angular distribution which is a weighted average over the various cross sections involved in the energy transfer process. Thus, these multigroup cross sections have the same format for both neutrons and gamma rays. In addition, the generation of secondary gamma rays may be considered as just another group-to-group transfer. Therefore using multigroup cross sections, the logic of the random walk process (the process of being transported from one collision to another) is identical for both neutrons and gamma rays.

The use of multigroup cross sections in a Monte Carlo code means that the effort required to produce cross-section libraries is reduced. Coupled neutron gamma-ray sets are available from the Radiation Shielding Information Center at Oak Ridge National Laboratory.

Cross sections may be read in either the DTF-IV[1] format or ANISN[2] and DOT[3] format. The ANISN-DOT type may be in either fixed or free form. The auxiliary information giving the number of groups, elements, coefficients, etc., is used to produce the necessary probability tables needed by the random walk module. The possible transport cases that can be

treated are neutron only, gamma ray only, coupled neutron-gamma ray, gamma ray from a coupled set, and fission, with all of the above options for either a forward or adjoint case and for isotropic or anisotropic scattering up to a P_{16} expansion of the angular distribution. The option of storing the Legendre coefficients for use in a next-event estimator is also provided.

The solution of the forward or normal transport equation by Monte Carlo generally involves a solution for $\chi(P)$, the density of particles with phase space coordinates $\underline{P}$ leaving collisions. Quantities of interest are then obtained by summing the contributions over all collisions, and frequently over most of phase space. The equations solved are derived in Section 4.10 and are written as Eqs. (40) and (95).

In some cases, it is of interest to solve the adjoint problem. This requires solving a transport problem with the detector response as a source. The various relationships between the adjoint and forward quantities are derived in Section 4.10. The adjoint equations solved by MORSE are Eqs. (93) and (99). In utilizing these adjoint equations, the logic of the random walk is the same as the forward mode.

Input to MORSE is read in five separate modules: (1) walk; (2) cross section; (3) user; (4) source; and (5) geometry. The walk input is read in subroutines INPUT1 and INPUT2 and includes all variables needed for the walk process. The cross-section input is read in cross-section module subroutines XSEC, JNPUT, and READSG. The parameters needed to set aside storage are read in XSEC, the mixing parameters are read in JNPUT, and the actual cross sections are read by READSG. Cross sections may be either on card or on tape. Input information required for analysis of the histories is read by subroutine SCØRIN of the analysis package which is called from INPUT2. Since the source varies from problem to problem, input may also be read in by subroutine SØRIN for the definition of the source. The geometry input is read by subroutine JØMIN. Any additional input required by the user may be read in by subroutine INSCOR which is called from SCØRIN.

In general, output of input parameters occurs in the same routine in which the input was read. In addition, there are two routines (ØUTPT

and ØUTPT2) for the output of results of the random walk process. Output of analysis results is generally performed in subroutine NRUN, but may also be done in a user-written routine ENDRUN which is called by NRUN.

Figure 4.1 shows the hierarchy of subroutines for MØRSE. From this diagram it is possible to see the functions of the modules. The input section takes care of setting up all variables needed in the transport process. Note that initial calculations by the cross-section module stem from XSEC. The analysis portion of the code is interfaced with MØRSE through BANKR with several uses made of cross-section routines in making estimates of the quantity of interest. With the exception of output from the walk process, the rest of the code consists of subroutine calls by MØRSE. The geometry module is interfaced through GØMST and the source is interfaced through MSØUR. The diagnostic module is independent and any part of it may be executed from any routine.

The diagnostic module provides an easy means of printing out, in useful form, the information in the various labelled commons and any part of blank common. The IBM-360 version also has the following features: a special routine is provided for printing out the particle bank; by loading parts of core with a junk word, the diagnostic package can determine which variables have been used; a "repeating line" feature is also included.

The geometry module consists of the combinatorial geometry package (CG) which is described in Section 4.7. It is based on the MAGI combinatorial geometry[4,5] but the format was changed to fit the MØRSE format: e.g., subroutines JØMIN, LØØKZ, and NØRML had to be written. The Ø5R-type[6] geometries are no longer maintained because the CG is easier to use and will handle all cases.

An albedo scattering may be forced to occur at every entry into a specified medium. A sample subroutine is provided for specular reflection and a subroutine call is provided (ALBIN, called from XSEC) for reading and storing albedo data of any degree of complexity. Thus transport of particles may be carried out in parts of the problem and an albedo scattering treated for other parts of the problem.

Time dependence is included by keeping track of the chronological age of the particle. For neutrons the age is incremented by the time needed to travel the distance between collisions if it traveled at a velocity corresponding to the average energy of the group. Provision is made for inputting a thermal group velocity separately. Nonrelativistic mechanics are assumed. The age of secondary gamma rays is determined from the neutron age at the collision site and is incremented by determining the time required to travel between collisions at the speed of light. For fission problems the age of the parent is given to the daughters at birth.

There are several types of importance sampling techniques included in the code. The Russian roulette and splitting logic of ØSR is an option in MØRSE. Also the exponential transform is provided with parameters allowed as a function of energy and region. Source energy biasing is an option as well as energy biasing at each collision. In fission problems the fission weights may be renormalized as a function of an estimate of k so that the number of histories per generation remain approximately constant. If desired, all importance sampling may be turned off.

Some other general features include the ability to run problems without the use of magnetic tapes, the ability to terminate a job internally after a set elapsed c.p.u. time and obtain the output based on the number of histories treated up to that time, batch processing for the purpose of determining statistics for groups of particles, and a repeat run feature so that results for a time-dependent fission problem may be obtained with statistical estimates. The output of numerous counters permits one to obtain an insight into the physics of the problem.

Descriptions of the subroutines are found on the following pages. Detailed derivations of various forms of the transport equation and a derivation of the treatment of the angular distribution of scattering are also included.

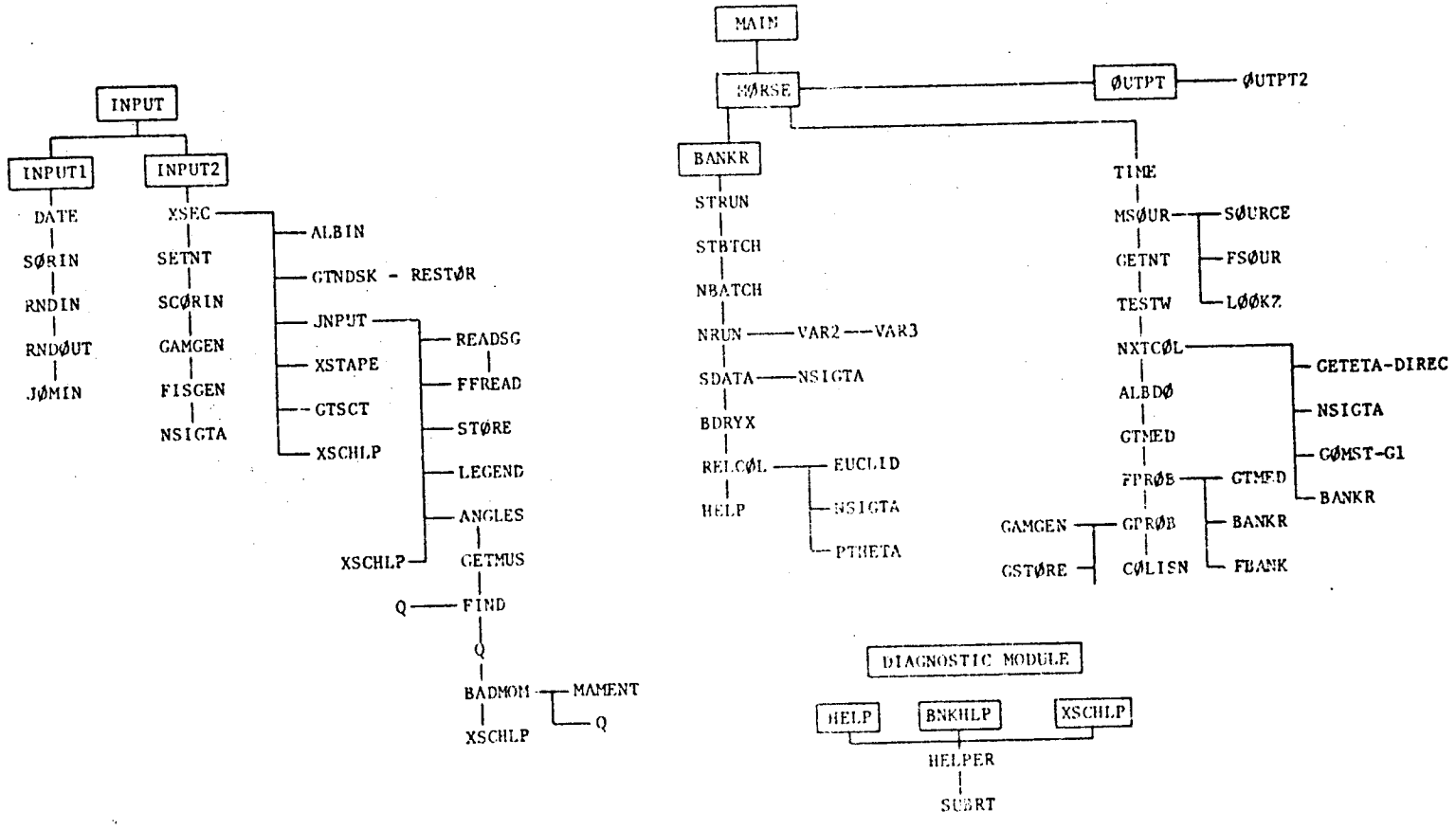


Fig. 4.1. Hierarchy of Subroutines in the MØRSE Code.

4.3. INPUT INSTRUCTIONS

4.3.1. Random Walk Input Instructions

The input read by subroutine INPUT1 is as follows:

CARD A (20A4)

Title card.

(Any character other than a blank or alphameric in column one will terminate the job.)

CARD B (10I5,F5.0,2I5)

- NSTRT - number of particles per batch.
- NMØST - maximum number of particles allowed for in the bank(s); may equal NSTRT if no splitting, fission, and secondary generation.
- NITS - number of batches.
- NQUIT - number of sets of NITS batches to be run without calling subroutine INPUT.
- NGPQTN* - number of neutron groups being analyzed.
- NGPQTG* - number of gamma-ray groups being analyzed.
- NMGP* - number of primary particle groups for which cross sections are stored; should be same as NGP (or the same as NGG when NGP = 0) on Card XB read by subroutine XSEC.
- NMTG - total number of groups for which cross sections are stored; should be same as NGP+NGG as read on Card XB read by subroutine XSEC.
- NCØLTP - set greater than zero if a collision tape is desired; the collision tape is written by the user routine BANKR.
- IADJM - set greater than zero for an adjoint problem.
- AXTIM - maximum clock time in minutes allowed for the problem to be on the computer (360/91 c.p.u. time); e.g., 4.5 entered here allows 4 and 1/2 minutes.
- MEDIA - number of cross-section media; should agree with NMED on Card XB read by subroutine XSEC.

---

*See Table 4.1 for sample input.

Table 4.1. Sample Group Input Numbers for Some Representative Problems*

Case A - Neutron Only Cross Sections (22 groups)

Case B - Gamma-Ray Only Cross Sections (18 groups)

Case C - Neutron-Gamma-Ray-Coupled Cross Sections (22-18 groups)

Input Variable	Problem Type: Case A Top 14 groups	Case B Top 17 groups	Case C Neutrons Only Top 14 groups	Case C Gamma Rays Only Top 17 groups	Case C Neutron-Gamma Top 14 Neutron, Top 17 Gamma Groups	
MORSE Input:						
NGPQTN	14	0	14	0	14	
NGPQTG	0	17	0	17	17	CARD B,
NMGP	22	18	22	18	22	Variables
NMTG	22	18	22	18	40	
NGP	22	18	22	0	22 †	
NGG	0	0	0	18	18	CARD XB
INGP	22	18	40	40	40	Variables

*For cross sections with full downscatter, NDS = NGP, NDSG = NGG, INDS = INGP, and ITBL = number of downscatters + number of upscatters + 3. Usually, ISGG = number of upscatters + 4; i.e., NUS + 4.

† Must be = to total number of neutron groups in the data - otherwise it picks up gammas from wrong location.

MEDALB - albedo scattering medium is absolute value of MEDALB; if MEDALB = 0, no albedo information to be read in, MEDALB < 0, albedo only problem - no cross sections are to be read, MEDALB > 0, coupled albedo and transport problem.

CARD C (4I5,5E10.5)

ISØUR - source energy group if > 0, if ISØUR < 0 or if ISØUR = 0 and NGPFS ≠ 0, SØRIN is called for input of Cards E1 and E2.

NGPFS - number of groups for which the source spectrum is to be defined. If ISØUR < 0, NGPFS ≥ 2.

ISBIAS - no source energy biasing if set equal to zero; otherwise the source energy is to be biased, and Cards E2 are required.

NØTUSD - an unused variable.

WTSTRT - weight assigned to each source particle.

EBØTN - lower energy limit of lowest neutron group (eV) (group NMGP).

EBØTG - lower energy limit of lowest gamma-ray group (eV) (group NMTG).

TCUT - age in sec at which particles are retired; if TCUT = 0, no time kill is performed.

VELTH - velocity of group NMGP when NGPQTN > 0; i.e., thermal-neutron velocity (cm/sec).

CARD D (7E10.4)

XSTRT, YSTRT, ZSTRT - coordinates for source particles.

AGSTRT - starting age for source particles.

UINP, VINP, WINP - source particle direction cosines if all are zero, isotropic directions are chosen.

Source data on Cards C and D will be overridden by any changes in subroutine SØURCE.

CARDS E1 (7E10.4) (Omit if ISØUR on Card C > 0 or if ISØUR = NGPFS = 0)

NGPFS values of FS, where FS equals the unnormalized fraction of source particles in each group.

CARDS E2 (7E10.4) (Omit if ISØUR > 0 or if ISØUR ≤ 0 and ISBIAS = 0)

If ISBIAS > 0, NGPFS values of BFS, the relative importance of a source in group I, are required.

CARDS F (7E10.4)

NMTG values of ENER, the energies (in eV) at the upper edge of the energy group boundaries.

NOTE: The lower energies of groups NMGP and NMTG were read on Card C.

CARD G (2I5,5X,36I1,5X,13I1) (Omit if NCØLTP on Card B ≤ 0)

NHISTR - logical tape number for the first collision tape.

NHISMX - the highest logical number that a collision tape may be assigned.

NBIND(J), J=1, 36 - an index to indicate the collision parameters to be written on tape.

NCØLLS(J), J=1, 13 - an index to indicate the types of collisions to be put on tape.

(See Tables 4.2 and 4.22 for information concerning NBIND and NCØLLS.)

CARD H (Z12), on IBM-360; (Ø20) on CDC-6600; (4X,Ø12) on UNIVAC-1108

RANDØM - starting random number.

CARD I (7I5)

NSPLT - index indicating that splitting is allowed if > 0.

NKILL - index indicating that Russian roulette is allowed if > 0.

NPAST - index indicating that exponential transform is invoked if > 0 (subroutine DIREC required).

NØLEAK - index indicating that non-leakage is invoked if > 0.

IEBIAS - index indicating that energy biasing is allowed if > 0.

MXREG - number of regions described by geometry input (will be set to one if ≤ 0).

MAXGP - group number of last group for which Russian roulette or splitting or exponential transform is to be performed. For adjoint, set = NMTG or overstoring results.

Table 4.2. Variables That May Be Written on Tape (NBIND)

J	Variable*	J	Variable
1	NCØLL	19	WTBC
2	NAME	20	ETAUSD
3	IG	21	ETA
4	U	22	AGE
5	V	23	ØLDAGE
6	W	24	NREG
7	X	25	NMED
8	Y	26	NAMEX
9	Z	27	WATEF
10	WATE	28	BLZNT
11	IGØ	29	BLZØN
12	UØLD	30	VEL(IG)
13	VØLD	31	VEL(IGØ)
14	WØLD	32	TSIG
15	XØLD	33	PNAB
16	YØLD	34	NXTRA
17	ZØLD	35	EXTRA1
18	ØLDWT	36	EXTRA2

*These variables are defined in Table 4.4 and Table 4.5.

CARD J (6I5,4E10.5) (Omit if NSPLT + NKILL + NPAST = 0)

NGP1, NDG, NGP2 — from energy group NGP1 to energy group NGP2, inclusive, in steps of NDG and from region NRG1 to NRG2, inclusive, in steps of NDRG, the following weight standards and path-stretching parameters are assigned. If NGP1 = 0, groups 1 to MAXGP will be used; if NRG1 = 0, regions 1 to MXREG will be used (both in steps of one). Usually NDG = 1 and NDRG = 1.

NRG1, NDRG, NRG2 — (see above)

WTHIH1 - weight above which splitting will occur.

WTLØW1 - weight below which Russian roulette is played.

WTAVE1 - weight given those particles surviving Russian roulette.

PATH - path-length stretching parameters for use in exponential transform (usually $0 \leq PATH < 1$).

The above information is repeated until data for all groups and regions are input.

End Cards J with negative value of NGP1 (ex., -1 in columns 4 and 5).

CARDS K (7E10.4) (Omit if IEBIAS on Card 1 ≤ 0).

((EPRØB(IG,NREG), IG = 1, NMTG), NREG = 1, MXREG)

Values of the relative energy importance of particles leaving a collision in region NREG. Input for each region must start on a new card.

CARD L (4I5)

NSØUR - set ≤ 0 for a fixed source problem; otherwise the source is from fissions generated in a previous batch.

MFISTP - index for fission problem, if ≤ 0 no fissions are allowed.

NKCALC - the number of the first batch to be included in the estimate of k; if ≤ 0 no estimate of k is made.

NØRMF - the weight standards and fission weights are unchanged if ≤ 0; otherwise fission weights will be multiplied, at the end of each batch, by the latest estimate of k and the weight standards are multiplied by the ratio of fission weights produced in previous batch to the average starting weight for the previous batch. For time-dependent decaying systems, NØRMF should be > 0.

CARDS M (7E10.4) (Omit if MFISTP on Card L $\leq$ 0)

(FWLØ(I), I = 1, MXREG) values of the weight to be assigned to fission neutrons.

CARDS N (7E10.4) (Omit if MFISTP on Card L $\leq$ 0)

(FSE(IG,IMED), IG = 1, NMTG), IMED = 1, MEDIA) the fraction of fission-induced source particles in group IG and medium IMED.

NOTE: Input for each medium must start on a new card.

CARDS O (7E10.5) (Omit if NGPQTN = 0 or NGPQTG = 0, i.e., include if coupled neutron-gamma-ray problem)

((GWLØ(IG,NREG) IG = 1, NMGP or NMTG - NMGP), NREG = 1, MXREG) - values of the probability of generating a gamma ray. NMGP groups are read for each region in a forward problem and NMTG-NMGP for an adjoint. Input for each region must start on a new card.

4.3.2. Combinatorial Geometry Input Instructions

The combinatorial geometry input data is read by the JØMIN subroutine, except for the region volumes VNØR(I), which are read by the GTVLIN subroutine whenever IVØPT = 3. For clarity of terminology, the terms "regions" and "media" have essentially the same meaning as in the ØSR Geometry Package, but are constructed in a different manner. The term "zone" is the same as the "region" as defined in the original combinatorial geometry package. The term "body" has the same meaning as in the original combinatorial geometry package.

CARD CGA (2I5,10X,10A6)

IVØPT - option which defines the method by which region volumes are determined; if
IVØPT = 0, volumes set equal to 1,
IVØPT = 1, concentric sphere volumes are calculated,
IVØPT = 2, slab volumes (1-dim.) are calculated,*
IVØPT = 3, volumes are input by card.

IDBG - if IDBG > 0, subroutine PR is called to print results of combinatorial geometry calculations during execution. Use only for debugging.

JTY - alphanumeric title for geometry input (columns 21-80).

*Not operational.

CARDS CGB (2X,A3,1X,I4,6D10.3)

One set of CGB cards is required for each body and for the END card (see Table 4.3). Leave columns 1-6 blank on all continuation cards.

ITYPE - specifies body type or END to terminate reading of body data (for example BØX, RPP, ARB, etc.). Leave blank for continuation cards.

IALP - body number assigned by user (all input body numbers must form a sequence set beginning at 1). If left blank, numbers are assigned sequentially. Either assign all or none of the numbers. Leave blank for continuation cards.

FPD(I) - real data required for the given body as shown in Table 4.3. This data must be in cm.

CARDS CGC (2X,A3,I5,9(A2,I5))

Input zone specification cards. One set of cards required for each input zone, with input zone numbers being assigned sequentially.

IALP - IALP must be a nonblank for the first card of each set of cards defining an input zone. If IALP is blank, this card is treated as a continuation of the previous zone card.

IALP - END denotes the end of zone description.

NAZ - total number of zones that can be entered upon leaving any of the bodies defined for this input region (some zones may be counted more than once). Leave blank for continuation cards for a given zone. (If NAZ $\leq$ 0 on the first card of the zone card set, then it is set to 5). This is used to allocate blank common.

Alternate IIBIAS(I) and JTY(I) for all bodies defining this input zone.

IIBIAS(I) - specify the "ØR" operator if required for the JTY(I) body.

JTY(I) - body number with the (+) or (-) sign as required for the zone description.

Table 4.3. Input Required on CGB Cards for Each Body Type

Card Columns / Body Type	ITYPE 3-5	IALP 7-10	Real Data Defining Particular Body 11-20	21-30	31-40	41-50	51-60	61-70	Number of Cards Needed
Box	BØX	IALP is assigned by the user or by the code if left blank.	Vx	Vy	Vz	H1x	H1y	H1z	1 of 2
			H2x	H2y	H2z	H3x	H3y	H3z	2 of 2
Right Parallelepiped	RPP		Xmin	Xmax	Ymin	Ymax	Zmin	Zmax	1
Sphere	SPH		Vx	Vy	Vz	R	-	-	1
Right Circular Cylinder	RCC		Vx	Vy	Vz	Hx	Hy	Hz	1 of 2
			R	-	-	-	-	-	2 of 2
Right Elliptic Cylinder	REC		Vx	Vy	Vz	Hx	Hy	Hz	1 of 2
			R1x	R1y	R1z	R2x	R2y	R2z	2 of 2
Ellipsoid	ELL		V1x	V1y	V1z	V2x	V2y	V2z	1 of 2
			L	-	-	-	-	-	2 of 2
Truncated Right Cone	TRC		Vx	Vy	Vz	Hx	Hy	Hz	1 of 2
			L1	L2	-	-	-	-	2 of 2
Right Angle Wedge	WED or RAW		Vx	Vy	Vz	H1x	H1y	H1z	1 of 2
			H2x	H2y	H2z	H3x	H3y	H3z	2 of 2

Table 4.3 (Cont'd.)

Card Columns Body Type	ITYPE 3-5	IALP 7-10	Real Data Defining Particular Body 11-20	21-30	31-40	41-50	51-60	61-70	Number of Cards Needed
Arbitrary Polyhedron	ARB		V1x	V1y	V1z	V2x	V2y	V2z	1 of 5
			V3x	V3y	V3z	V4x	V4y	V4z	2 of 5
			V5x	V5y	V5z	V6x	V6y	V6z	3 of 5
			V7x	V7y	V7z	V8x	V8y	V8z	4 of 5
			Face Descriptions (see note below)						5 of 5
Termination of Body Input Data	END								

NOTE: Card 5 of the arbitrary polyhedron input contains a four-digit number for each of the six faces of an ARB body. The format is 6D10.3, beginning in column 11. See the ARB write-up in Section 4.7 for an example.

CARDS CGD (14I5)

MRIZ(I) - MRIZ(I) is the region number in which the "Ith" input zone is contained (I = 1, to the number of input zones). Region numbers must be sequentially defined from 1.

CARDS CGE (14I5)

MMIZ(I) - MMIZ(I) is the medium number in which the "Ith" input zone is contained (I = 1, to the number of input zones). Medium numbers must be sequentially defined from 1.

CARDS CGF (7D10.5) (Omit if IVØPT ≠ 3)

VNØR(I) - volume of the "Ith" region (I = 1 to MXREG, the number of regions).

4.3.3. MØRSEC - Cross-Section Module Input Instructions

CARD XA (20A4)

Title card for cross sections. This title is also written on tape if a processed tape is written; therefore, it is suggested that the title be definitive.

CARD XB (13I5)

NGP* - the number of primary groups for which there are cross sections to be stored. Should be same as NMGP input in MØRSE.

NDS - number of primary downscatters for NGP (usually NGP).

NGG* - number of secondary groups for which there are cross sections to be stored.

NDSG - number of secondary downscatters for NGG (usually NGG).

INGP* - total number of groups for which cross sections are to be input.

ITBL - table length, i.e., the number of cross sections for each group (usually equal to number of downscatters + number of upscatters + 3).

ISGG - location of within-group scattering cross sections (usually equal to number of upscatters + 4).

NMED - number of media for which cross sections are to be stored - should be same as MEDIA input in MØRSE.

---

*See Table 4.1 for sample input.

NELEM - number of elements for which cross sections are to be read.

NMIX - number of mixing operations (elements times density operations) to be performed (must be ≥ 1).

NCØEF - number of coefficients for each element, including P_0.

NSCT - number of discrete angles (usually $NCØEF/2_{integral}$).

ISTAT - flag to store Legendre coefficients if greater than zero.

CARD XC (11I5)

IRDSG† - switch to print the cross sections as they are read if > 0.

ISTR† - switch to print cross sections as they are stored if > 0.

IFMU† - switch to print intermediate results of μ's calculation if > 0.

IMØM† - switch to print moments of angular distribution if > 0.

IPRIN† - switch to print angles and probabilities if > 0.

IPUN† - switch to print results of bad Legendre coefficients if > 0.

IDTF† - switch to signal that input format is DTF-IV format if > 0; otherwise, ANISN format is assumed.

IXTAPE - logical tape unit if binary cross section tape, set equal to 0 if cross sections are from cards. If negative, then the processed cross sections and other necessary data from a previous run will be read; in this case (IXTAPE < 0) no cross sections from cards and no mixing cards may be input. The absolute value of IXTAPE is the logical tape unit.

JXTAPE - logical tape unit of a processed cross-section tape to be written. This processed tape will contain the title card, the variables from common LØCSIG and the pertinent cross sections from blank common

†Switches are ignored if IXTAPE < 0.

IØ6RT - logical tape unit of a point cross-section tape in 06R format.

IGQPT - last group (MØRSE multigroup structure) for which the 06R point cross sections are to be used ($\leq$ NMGP).

CARD XD (14I5) (Omit if IXTAPE $\leq$ 0)

Element identifiers for cross-section tape. If element identifiers are in same order as elements on tape, the efficiency of the code is increased due to fewer tape rewinds.

CARDS XE (Omit if IXTAPE $\neq$ 0)

If cross sections are in free-form, a card with ** in columns 2 and 3 must precede the actual data.

ANISN format if IDTF $\leq$ 0; otherwise, DTF-IV format. Cross sections for INGP groups with a table length ITBL for NELEM elements each with NCØEF coefficients.

CARDS XF (2I5,E10.5) (Omit if IXTAPE < 0)

NMIX (see Card XB) cards are required.

KM - medium number.

KE - element number occurring in medium KM (negative value indicates last mixing operation for that medium). Failure to have a negative value causes code not to generate angular probabilities for that media (LEGEND and ANGLE not called).

RHØ - density of element KE in medium KM.

CARDS XG (I5) (Omit if IØ6RT $\leq$ 0)

NXPM - number of point cross-section sets per medium found on an 06R[7,8] tape.

= 1, total cross section only,

= 2, total + scattering cross section,

= 3, total, scattering, and ν*fission cross section.

NOTE: Cross sections and cross-section input data may be checked independently of MØRSE utilizing XCHEKR.[9] The input to XCHEKR consists of the cross-section cards XA through XG preceded by a card as follows:

Format (4I5)

IADJM - set greater than zero for an adjoint problem.

MEDIA - number of cross-section media; should equal NMED on Card XB.

NMGP - number of primary particle energy groups for which cross sections are to be stored; should equal NGP on Card XB.

NMTG - total number of energy groups for which cross sections are to be stored. Should be equal to INGP on Card XB.

4.3.4. SAMBØ Analysis Input Instructions

The following data are read from cards by SCØRIN:

CARD AA (20A4)

Title information - will be immediately output.

CARD BB (8I5)

ND - number of detectors (set = 1 if $\leq$ 0).

NNE - number of primary particle (neutron) energy bins to be used (must be $\leq$ NE).

NE - total number of energy bins (set = 0 if $\leq$ 1).

NT - number of time bins for each detector (may be negative, in which case |NT| values are to be read and used for every detector) (set = 0 if |NT| $\leq$ 1).

NA - number of angle bins (set = 0 if $\leq$ 1).

NRESP - number of energy-dependent response functions to be used (set = 1 if $\leq$ 0).

NEX - number of extra arrays of size NMTG to be set aside (useful, for example, as a place to store an array of group-to-group transfer probabilities for estimator routines).

NEXND - number of extra arrays of size ND to be set aside (useful, for example, as a place to store detector-dependent counters).

CARDS CC (3E10.4) (ND cards will be read)

X,Y,Z - detector location. (If other than point detectors are desired, the point locations must still be input and can be combined with additional data built in to user routines to fully define each detector.)

Note that the distance between the above points and the XSTRT, YSTRT, ZSTRT values and the initial age, AGSTRT, will be used to define the lower limit of the first time bin.

CARD DD (20A4)

Title or units for total responses for all detectors. Will be used in columns 54 through 133 of the title for the print of these arrays.

CARD EE (20A4)

Title or units for each total response for all detectors.

CARDS FF (7E10.4)

Response function values. NMTG values will be read in each set of FF cards. Input order is from energy group 1 to NMTG (order of decreasing energy).

NOTE: Cards EE and FF are read in the following order: EE, FF1, . . . FFN, EE, FF1, . . . FFN, etc. NRESP sets of EE, FF cards will be read.

CARD GG (20A4) (Omit if NE $\leq$ 1)

Units for energy-dependent fluence for all detectors.

CARDS HH (14I5) (Omit if NE $\leq$ 1)

Energy group numbers defining lower limit of energy bins (in order of increasing group number). The NNE (if > 0) energy must equal NGPQTN; the NE entry must be set to NMGP + NGPQTG for a combined problem, or else NGPQTG or NGPQTN.

CARD II (20A4) (Omit if |NT| $\leq$ 1)

Units for time-dependent total responses for all detectors.

CARD JJ (20A4) (Omit if |NT| $\leq$ 1 or NE $\leq$ 1)

Units for time and energy-dependent fluence for all detectors.

CARDS KK (7E10.4) (Omit if |NT| $\leq$ 1)

NT values of upper limits of time bins for each detector (in order of increasing time and detector number). The values for each detector must start on a new card. |NT| values only are read if NT is negative. They are then used for every detector.

CARD LL (20A4) (Omit if NA $\leq$ 1)

Units for angle- and energy-dependent fluence for all detectors.

CARD MM (7E10.4) (Omit if NA $\leq$ 1)

NA values of upper limits of angle bins (actually cosine bins; the NAth value must equal one).

Following the input for the SAMBØ analysis module, input cards for user-written routines INSCØR, SØURCE, and ENDRUN.

4.4. RANDOM WALK MODULE

4.4.1. Introduction

The basic random walk process of choosing a source particle and then following it through its history of events is governed by the routines in this module of MORSE. A given problem is performed by following a number of batches of particles which then constitute a run. Multiple runs are also permitted. The batch process feature is used so that statistical variations between groups of particles can be determined. Thus a batch of source particles is generated and stored in the bank. The random walk for this batch of particles is determined by picking one particle out of the bank and transporting it from collision to collision, splitting it into two particles, killing by Russian roulette, and generating secondary particles (either gamma rays or fission neutrons) and storing them in the bank for future processing. Termination of a history when a particle leaks from the system, reaches an energy cutoff, reaches an age limit, or is killed by Russian roulette.

The random walk module performs the necessary bookkeeping for the bank and the transportation and generation of new particles and relays this information to the analysis module for estimation of the desired quantities. Use is made of the cross-section module and the geometry module during the random walk process and the input-output routines for the reading and printing of pertinent information about the problem.

In this module the main program is used to set aside the storage required in blank common and to pass this information to subroutine MØRSE which is the executive routine for the random walk process. After performing the necessary input operations and setting up storage requirements, the walk process consists of three nested loops: one for runs, one for batches, and the inner-most is for particles. After each termination of the batch loop, some bookkeeping is required before the generation of a new batch of source particles. After the termination of a run, a summary of the particle terminations, scattering counters, and secondary production counters are output, as well as the results of Russian roulette and splitting for each group and region.

There are only two main labelled commons (APØLLØ and NUTRØN) in the random walk routines. Tables 4.4 and 4.5 list the definitions of the variables in these two commons. Note that in Table 4.5 "current" and "previous" refer to values of parameters leaving the current and previous event sites, respectively (WTBC is the exception, being the weight entering the current event site). Also note that "event" includes boundary crossings, albedo collisions, etc., as well as real collisions. A description of blank common is given in Fig. 4.2, along with definitions in Table 4.8. The locations of the variables are given in Table 4.9. All the variables used as location labels, except NGEØM, locate cell zero of an array. Cells NLAST + 1 to NLAST + NLEFT are available for analysis arrays if the user supplies his own analysis package.

A description of the subroutines that make up the random walk module is given on the following pages in this section.

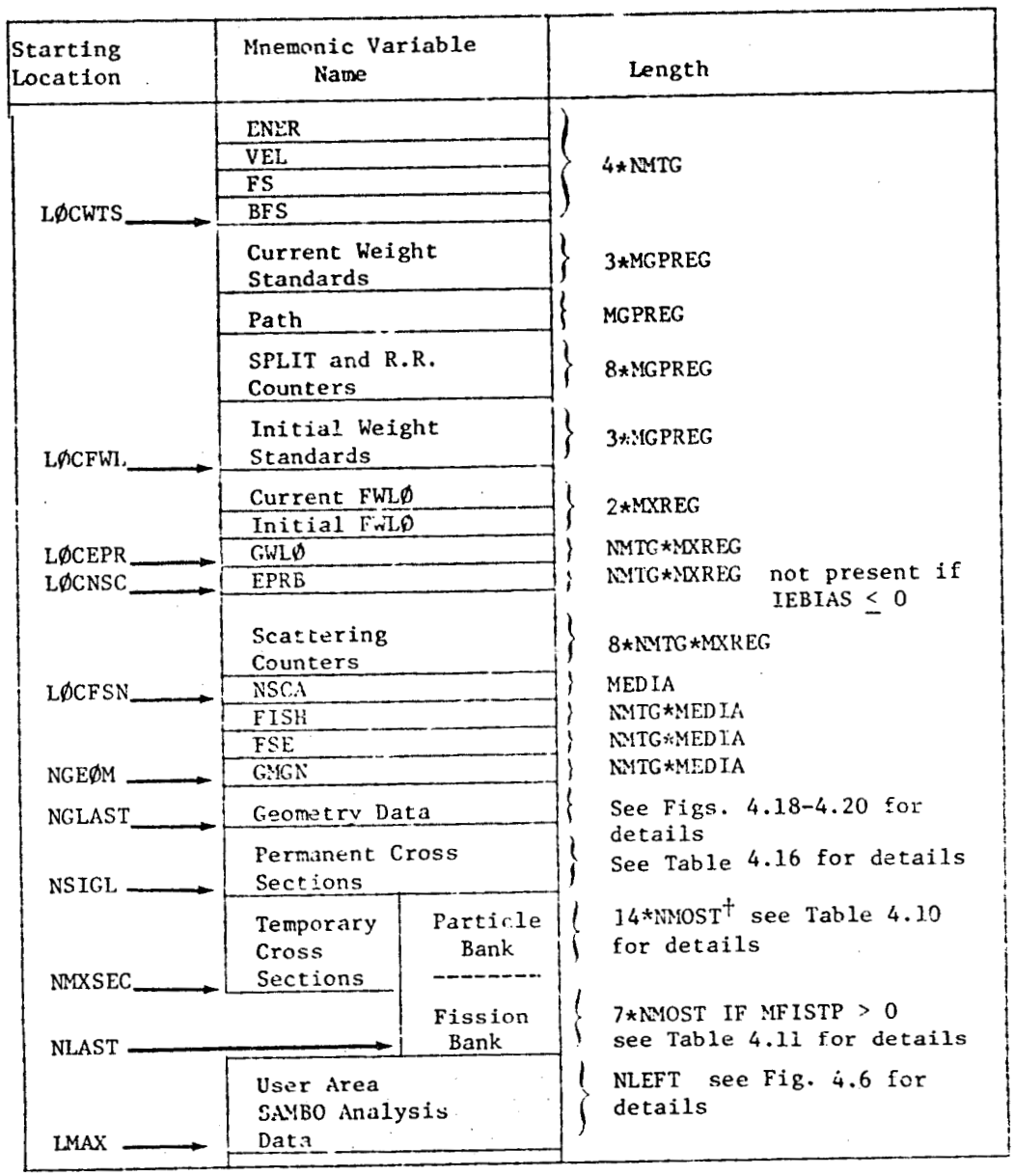


Fig. 4.2. General Layout of Blank Common.

†12*NMOST in IBM 360 version.

Table 4.4. Definition of Variables in Common APØLLØ

Variable	Definition
AGSTRT	Input starting age of source particle
DDF	Starting particle weight as determined in SØRIN
DEADWT(5)	The summed weights of the particles at death. The four deaths are: Russian roulette, escape, energy, and age limit. DEADWT(5) is unused.
ETA	Mean-free-path between collisions
ETATH	Distance in cm to the next collision if the particle does not encounter a change in total cross section.
ETAUSD	Flight path in m.f.p. that has been used since the last event
UINP, VINP, WINP	Input direction cosines for source particle
WTSTRT	Input starting weight
XSTRT, YSTRT, ZSTRT	Input starting coordinates for source particle.
TCUT	Age limit at which particles are retired.
XTRA(10)	Used for temporarily storing alphanumeric information on 'time used' immediately before printing it.
IO, I1	Output and input logical units
MEDIA	Number of media for which there are cross sections
IADJM	Switch indicating an adjoint problem if > 0.
ISBIAS	Switch indicating that source energy distribution is to be biased if > 0.

Table 4.4 (Cont'd.)

Variable	Definition
ISØUR	Input source energy group if > 0; SØRIN is called to read input spectrum if < 0; if = 0, SØRIN is called unless NGPFS = 0.
ITERS	Number of batches still to be processed in the run.
ITIME	Not used
ITSTR	Switch indicating that secondary fissions are to be the source for the next batch if > 0.
LØCWTS	Starting location in blank common of the weight standards and other arrays MGPREG long (see Fig. 4.2 and Table 4.9).
LØCFWL	Starting location in blank common of the fission weights.
LØCEPR	Starting location in blank common of the energy-biasing parameters.
LØCNSC	Starting location in blank common of the scattering counters
LØCFSN	Starting location in blank common of the fission and gamma-generation probabilities for each medium and group.
MAXGP	Maximum number of energy groups for which there are weight standards or path-length stretching parameters. For adjoint make MAXGP = NMTG or overstoring can occur.
MAXTIM	The elapsed clock time at which the problem is terminated.

Table 4.4 (Cont'd.)

Variable	Definition
MEDALB	Medium number for the albedo medium.
MGPREG	Product of number of weight standard groups (MAXGP) and regions (MXREG).
MXREG	Maximum number of regions in the system.
NALB	An index indicating that an albedo scattering has occurred if > 0.
NDEAD(5)	Number of deaths of each type (see DEADWT).
NEWNM	Name of the last particle in the bank.
NGEØM	Location of first cell of geometry data storage in blank common.
NGPQT1*	The lowest energy group (largest group number) for which primary particles are to be followed.
NGPQT2*	The number of primary particle groups.
NGPQT3*	The lowest energy group (largest group number) for which any particle is to be followed.
NGPQTC*	Number of energy groups of secondary particles to be followed.
NGPQTN*	Number of energy groups of primary particles to be followed.
NITS	Number of batches per run.

*See Fig. 4.3 for diagram of energy group structure.

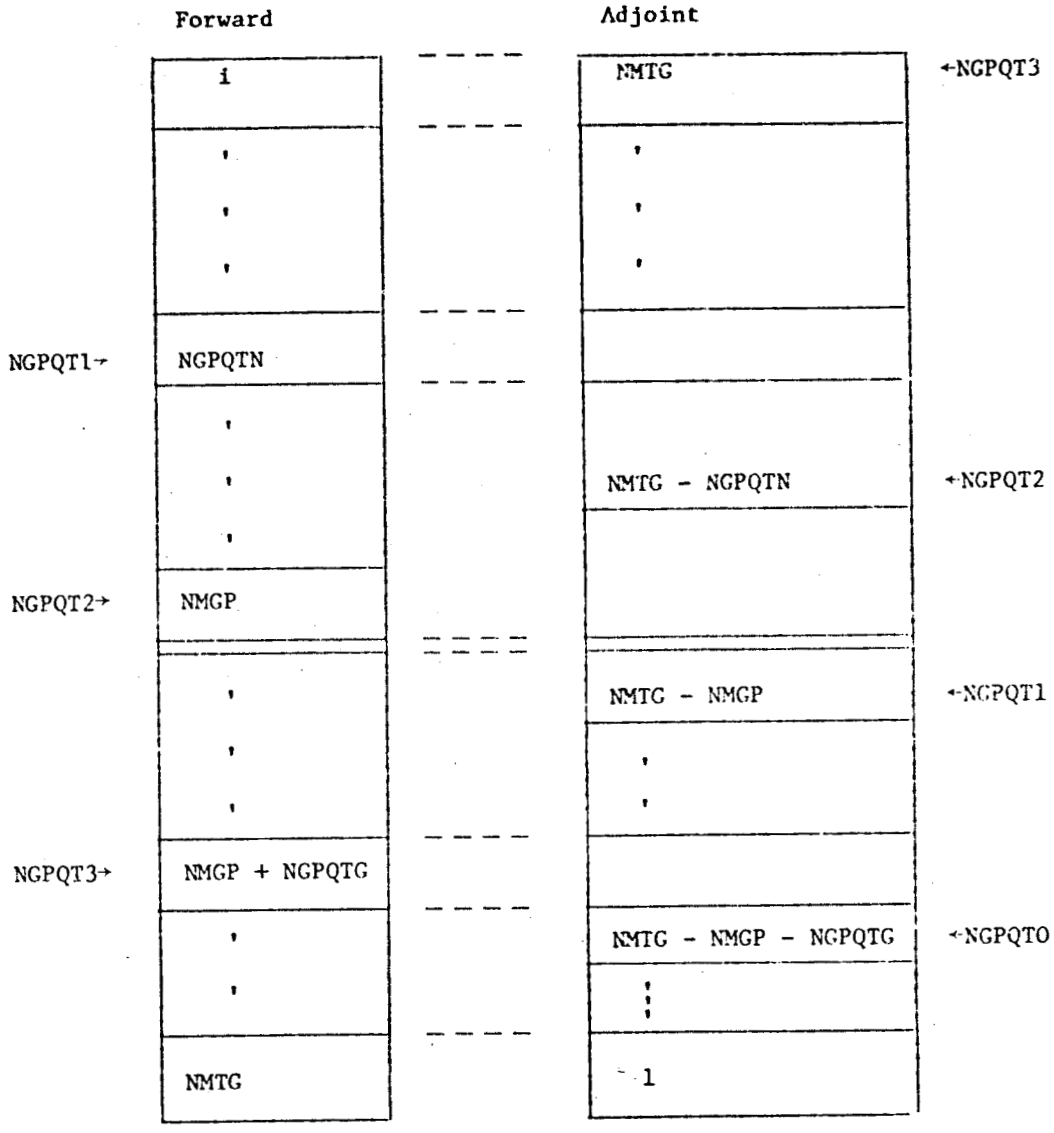
Table 4.4 (Cont'd.)

Variable	Definition
NKCALC	The first batch to be used for a k calculation. If 0, k is not calculated.
NKILL	An index to indicate that Russian roulette is to be played if > 0.
NLAST	The last cell in blank common that was used by the cross-section storage or is set aside for banking.
NMEM	The location of the next particle in the bank to be processed.
NMGP*	The number of primary particle groups for which there are cross sections.
NMØST	The maximum number of particles that the bank can hold.
NMTG*	The total number of energy groups (both primary and secondary) for which there are cross sections.
NØLEAK	An index which indicates that nonleakage path-length selection is to be used if > 0.
NØRMF	An index to indicate that the fission parameters are to be renormalized if > 0.
NPAST	An index to indicate that the exponential transform is to be used if > 0.
NPSCL(13)	An array of counters of events for each batch: (1) sources generated (2) splittings occurring

*See Fig. 4.3 for diagram of energy group structure.

Table 4.4 (Cont'd.)

Variable	Definition
	(3) fissions occurring
	(4) gamma rays generated
	(5) real collisions
	(6) albedo scatterings
	(7) boundary crossings
	(8) escapes
	(9) energy cutoffs
	(10) time cutoffs
	(11) Russian roulette kills
	(12) Russian roulette survivors
	(13) gamma rays not generated because bank was full
NQUIT	Number of runs still to be processed.
NSIGL	Starting location of the bank in blank common.
NSØUR	An index input to indicate that fissions are to be the source for future batches.
NSPLT	An index to indicate that splitting is to be considered if > 0.
NSTRT	The number of particles to be started in each batch.
NXTRA	NXTRA(1) used to transfer NLEFT from INPUT1 to INPUT2. NXTRA(2) used to transfer NGPQTO from INPUT1 to INPUT2.
NXTRA(3-10)	Not used.



NOTE: NGPQT1 = NGPQT3 for neutron only or gamma ray only.

Fig. 4.3. Diagram of Energy Group Structure

Table 4.5. Definition of Variables in NUTRON Common

Variable	Definition
NAME	Particle's first name.
NAMEX	Particle's family name. (Note that particles do not marry.)
IG	Current energy group index.
IGØ	Previous energy group index.
NMED	Medium number at current location
MEDØLD	Medium number at previous location.
NREG	Region number at current location.
U,V,W	Current direction cosine.
UØLD,VØLD, WØLD	Previous direction cosines
X,Y,Z	Current location.
XØLD,YØLD, ZØLD	Previous location.
WATE	Current weight.
ØLDWT	Previous weight. (Equal to WTBC if no path length stretching.)
WTBC	Weight just before current collision.
IBLZN	Current zone number.
IBLZØ	Previous zone number.
AGE	Current age.
ØLDAGE	Previous age.

Table 4.6. List of Routines Altering Variables in NUTRØN Common

Variable	Routine Where Variables May be Altered*
NAME	MSØUR, FSØUR
NAMEX	MSØUR, FSØUR
IG	MSØUR, SØURCE, FSØUR, CØLISN
IGØ	MSØUR, MØRSE
NMED	MSØUR, GØMST, FSØUR
MEDØLD	MSØUR, MØRSE, NXTCØL
NREG	MSØUR, GØMST, FSØUR
U,V,W	MSØUR, FSØUR, CØLISN
UØLD, VØLD, WØLD	MSØUR, MØRSE
X,Y,Z	MSØUR, GØMST, FSØUR
XØLD, YØLD, ZØLD	MSØUR, MØRSE, NXTCØL
WATE	MSØUR, SØURCE, TESTW, GETETA, GTIØUT, FSØUR, NXTCØL, CØLISN
ØLDWT	MSØUR, MØRSE, TESTW
WTBC	NXTCØL
IBLZN	MSØUR, GØMST, FSØUR
IBLZØ	MSØUR, MØRSE, NXTCØL
AGE	MSØUR, NXTCØL, FSØUR
ØLDAGE	MSØUR, MØRSE, NXTCØL

*Does not include routines where variables are temporarily altered but restored before returning to calling program.

Table 4.7. Subroutines in Which Variables in NUTRØN Common are Changed

Subroutine	Variables Changed
MØRSE	IGØ, UØLD, VØLD, WØLD, ØLDWT, XØLD, YØLD, ZØLD, IBLZØ, MEDØLD, ØLDAGE
MSØUR	all except WTBC
SØURCE	WATE, IG
(TESTW)	(WATE, ØLDWT)*
NXTCØL	WTBC, AGE, (XØLD, YØLD, ZØLD, IBLZØ, MEDØLD, ØLDAGE), (WATE)
GETETA	(WATE)
GØMST	X, Y, Z, NMED, NREG, IBLZN
(ALBDØ)	U, V, W
CØLISN	IG, U, V, W, WATE
(GTIØUT)	WATE

*Indicates possible change or call.

Table 4.8. Definitions of Variables in Random Walk Blank Common

Mnemonic Variable Name	Definition
ENER(IG)	Upper energy boundary of group IG (in eV).
VEL(IG)	Velocity corresponding to the mean energy for neutron groups and the speed of light for gamma-ray groups (in cm/sec).
FS(IG)	Unbiased source spectrum - unnormalized fraction of source particles in each energy group - transformed to c.d.f. by SØRIN.
BFS(IG)	Biased source spectrum - relative importance of each energy group - transformed to biased c.d.f. by SØRIN.
WTHI(IG, NREG)	Weight above which splitting is performed (vs. group and region).
WTLØ(IG, NREG)	Weight below which Russian roulette is performed (vs. group and region).
WTAV(IG, NREG)	Weight to be assigned Russian roulette survivors (vs. group and region).
PATH(IG, NREG)	Exponential transform parameters (vs. group and region).
NSPL(IG, NREG)	Splitting counter (vs. group and region).
WSPL(IG, NREG)	Weight equivalent to NSPL.
NØSP(IG, NREG)	Counter for full bank when splitting was requested (vs. group and region).
WNØS(IG, NREG)	Weight equivalent to NØSP.

Table 4.8 (Cont'd.)

Mnemonic Variable Name	Definition
RRKL(IG, NREG)	Russian roulette death counter (vs. group and region).
WRKL(IG, NREG)	Weight equivalent to RRKL.
RRSU(IG, NREG)	Russian roulette survival counter (vs. group and region).
WRSU(IG, NREG)	Weight equivalent to RRSU.
INIWHI (IG,NREG)	Initial values of WTHI array.
INIWLØ (IG,NREG)	Initial values of WTLØ array.
INIWAV (IG,NREG)	Initial values of WTAV array.
FWLØ(NREG)	Weights to be assigned to fission daughters (vs. region).
INIFLØ (NREG)	Initial values of FWLØ.
GWLØ(IG, NREG)	Weights to be assigned to secondary particles (vs. group and region).
EPRB(IG, NREG)	Relative importance of energy groups after scattering (vs. group and region). Present only if IEBIAS > 0.
NSCT(IG, NREG)	Number of real scatterings (vs. group and region).
WSCT(IG, NREG)	Weight equivalent to NSCT.
NALB(IG, NREG)	Number of albedo scatterings (vs. group and region).

Table 4.8 (Cont'd.)

Mnemonic Variable Name	Definition
WALB(IG, NREG)	Weight equivalent to NALB.
NFIZ(IG, NREG)	Number of fissions (vs. group and region).
WFIZ(IG, NREG)	Weight equivalent to NFIZ.
NGAM(IG, NREG)	Number of secondary productions (vs. group and region).
WGAM(IG, NREG)	Weight equivalent to NGAM.
NSCA(IMED)	Scattering counter (vs. cross-section medium).
FISH(IG, IMED)	Probability of generating fission neutron (vs. group and medium).
FSE(IG, IMED)	Source spectrum for fission-induced neutrons for each group - input as frequency of group IG.
GMGN(IG, IMED)	Probability of generating secondary particle (vs. group and medium).

Table 4.9. Indexing of Random Walk Blank Common Arrays

Mnemonic Variable Name	Location of Array in Blank Common (BC(I) or NC(I))
ENER(IG)	BC(I); I = IG
VEL(IG)	I = NMTG + IG
FS(IG)	I = 2*NMTG + IG
BFS(IG)	I = 3*NMTG + IG
WTHI(IG,NREG)	BC(I); I = LØCWTS + (NREG-1)*MAXGP + IG
WTLØ(IG,NREG)	I = LØCWTS + MGPREG + (NREG-1)*MAXGP + IG
WTAV(IG,NREG)	I = LØCWTS + 2*MGPREG + (NREG-1)*MAXGP + IG
PATH(IG,NREG)	I = LØCWTS + 3*MGPREG + (NREG-1)*MAXGP + IG
NSPL(IG,NREG)	NC(I); I = LØCWTS + 4*MGPREG + (NREG-1)*MAXGP + IG
WSPL(IG,NREG)	I = LØCWTS + 5*MGPREG + (NREG-1)*MAXGP + IG
NØSP(IG,NREG)	I = LØCWTS + 6*MGPREG + (NREG-1)*MAXGP + IG
WNØS(IG,NREG)	I = LØCWTS + 3*MGPREG + (NREG-1)*MAXGP + IG
RRKL(IG,NREG)	I = LØCWTS + 6*MGPREG + (NREG-1)*MAXGP + IG
WRKL(IG,NREG)	I = LØCWTS + 9*MGPREG + (NREG-1)*MAXGP + IG
RRSU(IG,NREG)	I = LØCWTS + 10*MGPREG + (NREG-1)*MAXGP + IG
WRSU(IG,NREG)	I = LØCWTS + 11*MGPREG + (NREG-1)*MAXGP + IG
INIWHI(IG,NREG)	BC(I); I = LØCWTS + 12*MGPREG + (NREG-1)*MAXGP + IG
INIWLØ(IG,NREG)	I = LØCWTS + 13*MGPREG + (NREG-1)*MAXGP + IG
INIWAV(IG,NREG)	I = LØCWTS + 14*MGPREG + (NREG-1)*MAXGP + IG
FWLØ(NREG)	BC(I); I = LØCFWL + NREG
INIFLØ(NREG)	I = LØCFWL + MXREG + NREG
GWLØ(IG,NREG)	I = LØCFWL + 2*MXREG + (NREG-1)*NMTG + IG
EPRB(IG,NREG)	BC(I); I = LØCEPR + (NREG-1)*NMTG + IG

Table 4.9 (Cont'd.)

Mnemonic Variable Name	Location of Array in Blank Common (BC(I) or NC(I))
NSCT(IG,NREG)	NC(I); I = LØCNSC + (NREG-1)*NMTG + IG
WSCT(IG,NREG)	BC(I); I = LØCNSC + NMTG*MXREG + (NREG-1)*NMTG + IG
NALB(IG,NREG)	NC(1); I = LØCNSC + 2*NMTG*MXREG + (NREG-1)*NMTG + IG
WALB(IG,NREG)	BC(I); I = LØCNSC + 3*NMTG*MXREG + (NREG-1)*NMTG + IG
NFIZ	NC(I); I = LØCNSC + 4*NMTG*MXREG + (NREG-1)*NMTG + IG
WFIZ	BC(I); I = LØCNSC + 5*NMTG*MXREG + (NREG-1)*NMTG + IG
NGAM	NC(I); I = LØCNSC + 6*NMTG*MXREG + (NREG-1)*NMTG + IG
WGAM	BC(I); I = LØCNSC + 7*NMTG*MXREG + (NREG-1)*NMTG + IG
NSCA(IMED)	NC(I); I = LØCNSC + IMED + 8*NMTG*MXREG
	or I = LØCFSN - MEDIA + IMED
FISH(IG,IMED)	BC(I); I = LØCFSN + (IMED-1)*NMTG + IG
FSE(IG,IMED)	I = LØCFSN + NMTG*MEDIA + (IMED-1)*NMTG + IG
GMGN(IG,IMED)	I = LØCFSN + 2*NMTG*MEDIA + (IMED-1)*NMTG + IG

Table 4.10. Layout of Particle Bank in Blank Common

Location	Variable	Comments
NNØ = NSIGL	IG	On IBM-360 this is a 2-byte word stored in last half of word.
	U	
	V	
	W	
	X	
	Y	Variables are from NUTRØN common, see Table definitions
	Z	
	WATE	
	AGE	
	BLZNT	
	NAME, NAMEX	On IBM-360 these are 2-byte words stored as one 4-byte word
	NMED, NREG	On IBM-360 these are 2-byte words stored as one 4-byte word
NNØ + 14		On IBM-360 this is NNØ+12
	Repeat for Particle 2	
	.	
	.	
NNØ + 14*NMØST = NLAST		On IBM-360 this is NNØ+12*NMØST
= NFISBN if MFISTP > 0		

4.4.2. Main Program

The main program performs the following functions:

1. Sets the maximum allowed size of blank common (all other routines using blank common use a dummy dimension of 1);
2. Ensures that certain labelled commons are loaded in a specified order (which must agree with the order of these commons in the diagnostic routines using the LØC function);*
3. Loads the junk word into blank common on all machines and into all labelled commons present in this routine on the IBM-360. The junk word is (48484848_{16}) on the IBM-360, (474747474747_8) on the UNIVAC, and 3000777777000000000B on the CDC-6600;
4. Sets the two variables used for input and output logical units; and
5. Calls MØRSE for the actual administration of the job. (The size of blank common is transferred to MØRSE as an argument.)

Subroutines called: MØRSE

Functions required: LØC (library function at Oak Ridge National Laboratory - output is the absolute address (in 8-bit bytes) of the cell given as the argument). At most installations other than ORNL, user must set NLFT to the dimension in blank common and not use LØC.

Variables required: JUNK

Variables changed: ITØUT (IO in most other routines).
ITIN (I1 in most other routines).

Commons required: Blank, APØLLØ, FISBNK, NUTRØN, LØCSIG, MEANS, MØMENT, QAL, RESULT, GEØMC, NØRMAL, PDET, USER, DUMMY.

Helpful Hints:

1. Note that if a new cross section, geometry, or analysis package is used, the labelled commons here may have to be modified correspondingly.

---

*This does not apply to non IBM-360 versions.

2. The junk word is the bit pattern that comes closest to being output identically as either a fixed or floating number. It is also recognized by subroutine HELPER that the cell has not been used by the code.
3. The LØC function returns an absolute address of a variable in bytes, requiring division by 4 to obtain the number of 4-byte (32 bit) words.†
4. To change the size of blank common only the statement defining the common needs to be changed at ORNL since the LØC function is used to obtain this value to be transferred to MØRSE; however, NLFT must also be changed at most other installations.
5. It is recommended that this routine always be compiled and that it be the first routine compiled. This insures that it is loaded first and that the commons it specifies are loaded first.
6. The program size, in bytes, is usually on the order of 155,000 + 4*(blank common size in words). Note: this does not include library routines.

---

†See Section 4.9 for a description of all library routines used in MØRSE.

4.4.3. Subroutines

Subroutine MØRSE (NLFT)

MØRSE is the executive routine for the walk process and controls the succession of events which comprise the Monte Carlo process. The problem is assumed to consist of NQUIT runs, each consisting of NITS batches, and starting out with NSTRT particles in each batch. Thus the functions of MØRSE are logically broken down into nested loops with the inner loop consisting of the execution of the walk process for each particle. The next loop is for each batch of particles and the outer loop is for each run. Several problems may be run in succession by stacking input data.

There is no significant part of the walk process performed in MØRSE except for the termination of histories. The bookkeeping of before-collision parameters, the determination of history terminations, and the ordering of the subroutine calls are the basic functions. The option of terminating a problem by an execution time limit is provided; this option may only be executed at the end of a batch and the normal termination of a problem occurs in that all end-of-run processes are completed.

Called from: Main program.

Subroutines called: INPUT, TIMER, BANKR(-1), BANKR(-2), MSOUR, ØUTPT(1), GETNT, TESTW, NXTCØL, BANKR(10)†, ALBDØ, BANKR(6), GTMED, FPRØB, GPRØB, CØLISN, BANKR(5), BANKR(9)†, BANKR(-3), ØUTPT(2), BANK(-4), ØUTPT(3), ICLØCK, RNDØUT

Commons required: Blank, APØLLØ, NUTRØN, FISBNK.

Variables required: NLFT, NKILL, NSPLT, NGPQTN, NGPQTG, NITS, NQUIT, NSTRT, NFISH, ITSTR, NMEM, MAXTIM, TCUT — from common APØLLØ (see page 4.4-4)

NALB - index indicating that an albedo collision has occurred.

MFISTP - index indicating that fissions are allowed if > 0.

---

†If user wishes to implement these calls, he must insert them.

Variables changed:

NDEAD(I), DEADWT(I) - counters

I = 1 - Russian roulette kill

= 2 - particle escaped the system

= 3 - particle reached energy cutoff

= 4 - particle reached age limit - it was retired.

NPSCL(I) - counters

I = 5 - number of real collisions

= 6 - number of albedo collisions

= 9 - number of energy deaths

= 10 - number of age terminations.

Subroutine DATE (A,NW) IBM-360 Version†

Given an array A, DATE inserts a hollerith string with the day of the week, the month, the day of the month, and the year. It will use as many as 32 bytes, so A must be dimensioned at 8 for single precision. NW, on return, is the number of 4-byte words which must be output.

Typical calling sequence for IBM-360 version:

```
     DIMENSIØN ARRAY (8)
     CALL DATE ( ARRAY, NUM)
     PRINT 1, (ARRAY(I),I=1,NUM)
   1 FØRMAT ('TØDAY IS ',8A4)
```

producing, if called on May 30, 1970:

'TØDAY IS SATURDAY, MAY 30, 1970'.

Called from: INPUT1

Routines called:

IWEEK

INTØBC (library function at Oak Ridge National Laboratory, converts a 4-byte integer to an EBCDIC string),

INTBCD - same as INTØBC except also returns the number of bytes in the EBCDIC string.

Commons: DATDAT which contains arrays of EBCDIC characters for months and weekdays, arrays of numbers of EBCDIC characters and starting points. It is loaded in a Block Data routine with the following values:

---

†The CDC-6600 version is a dummy. The UNIVAC-1108 version is described on page 4.4-24 of this section.

CØMMØN /DATDAT/ XMØNTH(11), WEKE(6), DAY(1), IMØNTH(12), NMØNTH(12), IWEKE(8), IWEEK(8)

Index	XMØNTH (REAL*8)	WEKE (REAL*8)	DAY (REAL*8)	IMØNTH	NMØNTH	IWEKE	IWEEK
1	JANUARY ① *	HUH?SUN ①	DAY, ① 19 ①	0	7	0	4
2	FEBRUARY	MØN ① TUES		8	8	4	3
3	MARCH ③	WEDNES ②		16	5	8	3
4	APRIL ③	THURS ③		24	5	12	4
5	MAY ① JUNE	FRI ⑤		32	3	16	6
6	JULYAUGU	SATUR ③		36	4	24	5
7	ST ② SEPT			40	4	32	3
8	EMBER ③			44	6	40	5
9	ØCTØBER ①			52	9		
10	NØVEMBER			64	7		
11	DECEMBER			72	8		
12				80	8		

* Ⓝ denotes N blanks.

Subroutine DATE (IO) - UNIVAC-1108 Version

This routine generates a line of print containing the month, day, year and time that the case begins.

Called from: INPUT1

Routines called:

ERTRAN - library routine that determines the date and time.

Variables required:

IO - logical unit number of st ndard output tape.

Function DIREC (Dummy)

This function must be provided by the user whenever the path-length stretching option is exercised. DIREC usually provides the cosine of the angle between the flight direction and the most important direction. It is used to vary the amount of path-length biasing depending on whether the particle is traveling in an important direction or not. The version that is distributed with the MØRSE code merely sets DIREC=1, which produces maximum path-length stretching when used in the calling routine GETETA.

Called from: GETETA

Variables changed:

DIREC - the function value.

Subroutine EUCLID (MRK, X1, Y1, Z1, X2, Y2, Z2, P1P2, IG, ARG, NTD, MEDIUM, IBLZ, NREGN)

This routine is provided for the user to determine the number of mean free paths between two points in the system. It will either return the total number of mean free paths or will return the first boundary intersection point and the number of mean free paths to that point. Total cross sections are determined by calling NSIGTA and distances to next interfaces are calculated by calling G1.

Called from: GETETA and user routines like RELCØL and SDATA.

Subroutines called: G1, NSIGTA.

Functions used: DSQRT or SQRT depending on precision of geometry (library).

Commons required: GØMLØC, PAREM, ØRGI, APØLLØ.

Variables required:

MRK	- Set to 1 upon calling. If NTD is non-zero, MRK should be 0 on successive calls for the same trajectory, so that new trajectory parameters are not initialized. This will be handled automatically if the calling routine does not change MRK.
X1, Y1, Z1	- Coordinates of starting point.
X2, Y2, Z2	- Coordinates of end point.
P1P2	- Distance between starting and end points.
IG	- Energy group index.
NTD	= 0 for total mean free paths, ≠ 0 for intersection points and mean free paths between.
MEDIUM	- Media in which X1, Y1, Z1 is located.
IBLZ	- Zone (IR) in which X1, Y1, Z1 is located.
NREGN	- Region in which X1, Y1, Z1 is located.

Variables changed:

MRK	= 1 for a flight reaching the end point, = 0 for a flight crossing a medium boundary (NTD ≠ 0 only), = -1 for a flight escaping the system, = -2 for a flight encountering an internal void (NTD ≠ 0 only).
X1, Y1, Z1	= Returns boundary intersection point if NTD ≠ 0.

ARG - Negative of number of mean free paths.

MEDIUM - Medium number of end point.

Significant internal variables:

MARK - Flag set (returned as MRK - defined above).

ETA - Distance remaining to end point in cm.

ETAUSD - Distance traveled on last call to G1.

Subroutine FBANK

Fission neutrons are banked in two different ways depending on the value of NSØUR. If NSØUR = 1, FBANK banks fission neutrons in the fission bank for the next batch. If NSØUR = 0, however, fission neutrons are banked in the source bank in a manner similar to secondary gamma-ray production. No neutrons are placed in the fission bank. NPSCL(4) is incremented with each fission and BANKR(4) is called for analysis of the neutrons produced by fission. In either case, beware if you use BANKR(3) - it may not have the appropriate information in NUTRØN common. Seven parameters can be stored for NMØST neutrons (see Table 4.11). If the bank is full, no particle is generated; and NPSCL(13) is incremented. An error message "Warning - no room in bank for secondaries" is printed the first 10 times it happens.

Called from: FPRØB.

Subroutines called: GTMED, GTISØ, STØRNT, BANKR(4), FLTRNF

Commons required: Blank, NUTRØN, APØLLØ, FISBNK

Variables required:

- NSØUR - see discussion above.
- NFISBN - location of cell zero of the fission bank in blank common.
- NFISH - number of neutrons in fission bank.
- NMØST - maximum number of particles allowed in bank.
- WATEF - weight of fission neutron to be banked.
- FWATE - total weight of banked fission neutrons.
- AGE, IG, NAMEX, X, Y, Z (from NUTRØN common, see page 4.4-10).

Variables changed:

- NFISH - incremented after banking.
- FWATE - incremented by WATEF after banking.
- NPSCL(4) - incremented if fission particle is banked.
- NPSCL(13) - incremented upon each call when bank is full.

Table 4.11. Layout of Fission Bank in Blank Common

Location	Variable
NFISBN	X
	Y
	Z
	WATEF
	AGE
	IG
	NAMEX
NFISBN + 7	
	Repeat for particle 2
	.
	.
	.
NFISBN + 7*NMØST = NLAST	

Table 4.12. Definitions of Variables in Common FISBNK

Variable	Definition
MFISTP	Index to indicate that fissions are to be treated if > 0.
NFISBM	Location in blank common of cell zero of the fission bank.
NFISH	Number of fissions produced in the previous batch (generation).
FTØTL	Total fission weight from all collisions in this batch (generation).
FWATE	Total weight of fission neutrons stored in bank in this batch (generation).
WATEF	Weight of fission neutron stored in bank.

Random Number Package

The random number package is essentially the Ø5R[6] package as modified for the IBM-360 computers. Six-byte (48 bit) arithmetic is used with a generator (constant multiplier) equal to $1AFD498D_{16}$ (= 3277244615_8). If no starting number is given (a value of zero input) the routine uses $35FA931A_{16}$ which is twice the generator. The trailing zero bit restricts the significance of the arithmetic to 47 bits so that the pseudo-random sequence generated by the CDC-1604 package may be duplicated. (The CDC-1604 package must use 3277244615_8 as the generator and starting number to give the same sequence.) The CDC-6600 version starts with a built-in generator of 000064722612673171611B.

The following subprograms are available in the package:

FORTRAN Calling Statement	Random Number Generated
R = FLTRNF (0)	Uniformly distributed on the interval (0,1).
R = SFLRAF (0)	Uniformly distributed on the interval (-1,1).
R = EXPRNF (0)	Exponentially distributed: $P(R)\,dR = e^{-R}dR$ $0 \leq R < \infty$.
CALL AZIRN (SIN,CØS)	The sine and cosine of ϕ where ϕ is uniformly distributed on the interval $(0,2\pi)$. A random azimuthal angle.
CALL PØLRN (SIN,CØS)	The sine and cosine of θ where $\cos\theta$ is uniformly distributed on the interval (-1,1). A random polar angle.
CALL ISTISØ (X,Y,Z)	An isotropic unit vector. $X = \cos\theta$, $Y = \cos\phi \sin\theta$, $Z = \sin\phi \sin\theta$ where θ is a random polar angle and ϕ is a random azimuthal angle.
R = RNMAXF(T)	Maxwellian energy: $P(R)\,dR = \left(\frac{4}{T^3\pi}\right)^{1/2} R^{1/2}\, e^{-R/T}\, dR.$

FØRTRAN Calling Statement	Random Number Generated
R = FISRNF (0)	A neutron speed squared from the Watt fission spectrum $P(R)\ dR = e^{-R/T} \sinh (2\sqrt{R*E_f}/T)$, where $T = 0.965 \times 1.913220092 \times 10^{18}$ and $E_f = 0.533 \times 1.913220092 \times 10^{18}$.†
CALL RNDIN(R) or (I)	The IBM-360 version loads R into RANDØM(I), I = 2, 4 if R ≠ 0. R is read with a Z12 format and must be double precision (8 bytes). The UNIVAC-1108 and CDC-6600 versions read with Ø formats and load I into NRANDM.
CALL RNDØUT(R) or (I)	Loads RANDØM(I), I = 2, 4 into R on IBM-360 or loads NRANDM into I on the other two systems.

NOTE: The arguments of FLTRNF, SFLRAF, EXPRNF, FISRNF are not used by the routines.

†L. Cranberg, et al., Phys. Rev. 103, 662 (1956).

Subroutine FPRØB

FPRØB calculates the expected weight of fission neutrons at a collision point and then splits or plays Russian roulette so as to produce the correct average number of fissions, all of weight FWLØ (specified in problem input for each region). FBANK is called for each neutron produced, to be stored for processing in the next generation. FPRØB calls FISGEN for forward fission problems. For adjoint problems, $\overline{\nu}\Sigma_f/\Sigma_t$ is selected from the FISH array in blank common.

Called from: MØRSE.

Subroutines called: FISGEN, GTMED, BANKR(3), FBANK, FLTRNF, HELP, ERRØR, SIGN (library).

Commons required: Blank, NUTRØN, APØLLØ, FISBNK.

Variables required:

NMED, WATE, NREG, IG (from NUTRØN common, see page 4.4-10)

IADJM - =1 if adjoint problem;
=0 if forward problem.

LØCFSN - location in blank common of cell zero of array of fission cross sections.

LØCNSC - location in blank common of cell zero of scattering counter arrays.

IMED - cross-section medium of collision point.

MXREG - maximum region number.

NMTG - total number of energy groups.

FTØTL - total of fission weights from all collisions.

LØCFWL - location in blank common of cell zero of array FWLØW.

NPSCL(3) - fission counter.

Variables changed:

WATEF - fission weight transferred to FBANK.

NFIZ - actual number of fissions per group and region.

WFIZ - weight equivalent to NFIZ.

FTØTL

NPSCL(3)

Significant internal variables:

FWL -current value from array FWLØ.

ISCT - location in blank common of (IG,NREG) cell of scattering counter array NFIZ (and later WFIZ).

Subroutine FSØUR

This routine is called by the source executive routine, MSØUR, when the source for the present batch is to be taken from the previous batch fissions. Its function is to transfer the neutron parameters from the fission bank to the neutron bank. If there were no fissions in the previous batch, it sets a flag, prints a message, and returns.

Called from: MSØUR

Subroutines called:

STØRNT(N) - loads parameters in common NUTRØN into the N^{th} location in the neutron bank.

Commons required: Blank, NUTRØN, FISBNK, APØLLØ.

Variables required:

NFISH - number of fissions produced in the previous batch.

NMEM - set equal to NFISH.

NITS - number of batches requested for the run.

ITERS - batch counter.

NFISBN - location in blank common of cell zero of the fission bank.

Variables changed:

NITS - set to number of batches completed if NFISH = 0.

ITERS - set to zero if NFISH = 0.

NAME, NMED, NREG, U, V, W, BLZNT — Set to zero (in NUTRØN common, see page 4.4-10).

X, Y, Z, WATE, AGE, IT, NAMEX — Set to values found in fission bank (in NUTRØN common, see page 4.4-10).

NOTE: IG is group index of neutron causing fission.

Subroutine GETETA

The subroutine GETETA selects ETA, the number of mean-free-paths for the next flight, from an appropriate exponential distribution. Path-length stretching based on the exponential transform[10-12] is included, as well as an option to select from a modified distribution which does not permit a particle to escape from the system.

The unbiased flight path distribution function is given by

$$P_0(\eta) = e^{-\eta}$$

where η is the distance traveled in mean-free paths. Selection of a particular flight path ETA from $P_0(\eta)$ is done by the function EXPRNF (in random number package, see page 4.4-31).

If an external boundary occurs at some distance, ARG mean-free paths from the starting point along the flight direction, then the probability of escape is e^{-ARG}. If it is required that no particle escape, then the distribution function $e^{-\eta}$ is normalized over the interval (0,ARG), and the flight path is selected from the modified distribution

$$P_1(\eta) = \frac{e^{-\eta}}{(1 - e^{-ARG})}$$

and the particle's weight is adjusted by the factor

$$\frac{P_0(\eta)}{P_1(\eta)} = (1 - e^{-ARG}).$$

Path-length stretching, which is a form of biasing (or importance sampling), can be accomplished by selecting from the modified distribution

$$P_2(\eta) = \frac{1}{BIAS} e^{-\eta/BIAS},$$

which produces values of ETA a factor of BIAS times those produced by the unbiased distribution $P_0(\eta)$. Therefore, values of BIAS greater than unity will stretch the path length and values less than unity will shrink the path length. The actual selection is accomplished in terms of the distribution function for $\eta' = \eta/BIAS$,

$$P_2(\eta') = P_2(\eta)\left|\frac{d\eta}{d\eta'}\right| = e^{-\eta'}.$$

A selection is made from $P_2(\eta')$ which yields values of ETA' and then

$$ETA = BIAS*ETA'.$$

If path-length biasing is used, then the particle's weight must be adjusted by the factor

$$\frac{P_0(ETA)}{P_2(ETA)} = BIAS\ e^{-[1\ -\ (1/BIAS)]*ETA}.$$

For the combination of path-length stretching and no escape, the modified distribution is given by

$$P_3(\eta) = \frac{e^{-\eta/BIAS}}{BIAS*(1\ -\ e^{-ARG/BIAS})}$$

with the actual selection of ETA' being made from the modified distribution

$$P_3(\eta') = P_3(\eta)\left|\frac{d\eta}{d\eta'}\right| = \frac{e^{-\eta'}}{(1\ -\ e^{-ARG/BIAS})}$$

where η = BIAS*η'. The path-length ETA is then given by

$$ETA = BIAS*ETA',$$

and the particle's weight multiplied by the factor

$$\frac{P_0(ETA)}{P_3(ETA)} = BIAS*e^{-ETA(1\ -\ 1/BIAS)}(1\ -\ e^{-ARG/BIAS}).$$

The form for the factor BIAS used in this version of GETETA is based on the exponential transform and can be expressed as

$$BIAS = \frac{1}{(1\ -\ PATH*DIREC)}$$

where

DIREC is the cosine of the angle between the flight direction and the most important direction (calculated by the user function DIREC),

PATH is a measure of the maximum amount of path-length stretching to be applied. A value of zero corresponds to BIAS = 1.0, and no biasing is accomplished. Larger values of PATH (but less than unity) yield values of BIAS > 1.0 when DIREC > 0, and the particle's path length is stretched accordingly. Conversely, when DIREC < 0 (the particle is traveling away from the important direction) BIAS < 1.0 and the track is shortened.

Called from: NXTCØL.

Subroutines called: EUCLID.

Function used: DIREC. EXPRNF, AMØD(library), EXP(library)

Commons required: Blank, NUTRØN, APØLLØ.

Variables required:

IG, X, Y, Z, U, V, W, WATE, NREG (from NUTRØN common, see page 4.4-10).

MAXGP - number of energy groups for weight standards and/or path-length stretching parameters PATH.

NØLEAK - an index for nonleakage biasing.

RAD - the largest overall dimension in the system.

PATH - path-length stretching parameters (in blank common).

Variables changed:

ETA - the number of mean-free paths to the next collision.

WATE - the particle's weight corrected for the biasing employed during the present flight selection.

Significant internal variables:

ARG - the distance in mean-free paths from the last collision site to an external boundary along the present flight direction.

Subroutine GETNT(N)

Three entry points are used in this routine.† Entry SETNT saves the address (in words) of the first cell available for the neutron bank in blank common and returns the address of the last cell it will use. Entry STØRNT(N) stores values from common NUTRØN into the N^{th} set of locations in the neutron bank and entry GETNT(N) does the reverse; it picks up variables from the bank and puts them in common NUTRØN.

Called from: INPUT2 (SETNT), MØRSE (GETNT), MSØUR (STØRNT), FSØUR(STØRNT), ØUTPT (GETNT).

Commons required: Blank, NUTRØN. The area of blank common used for the neutron bank is shown in Table 4.10. Notice that IG, NAME, NAMEX, NMED, and NREG are stored in 2-byte words (and are therefore limited to $\leq$ 65535) on the IBM-360, but are full words on the UNIVAC-1108 and CDC-6600.

Variables required:

SETNT: NLAST
NMØST

GETNT: N

STØRNT: N, NAME, NAMEX, NMED, NREG, IG, U, V, W, X, Y, Z, WATE, BLZNT, AGE (from NUTRØN common, see page 4.4-10).

Variables changed:

SETNT: NLAST

GETNT: variables in common NUTRØN required by SETNT above.

STØRNT: 12 locations in blank common.

Significant internal variables:

NNØ - location in blank common of start of neutron bank.

†On the CDC-6600 3 separate subroutines exist.

Subroutine GØMST (TSIG, MARK)

Any boundary crossings between the present and next collision sites are determined by calling the combinatorial geometry routine Gl. Before the Gl call, combinatorial geometry variables in common PAREM are initialized; and after the call, NUTRØN variables are updated. If an albedo medium is encountered, the flag NALB is set to the albedo medium number, and then NØRML is called to determine the normal to the surface encountered. MARK is set to -1 for an external void.

Called from: NXTCØL.

Subroutines called: Gl, NØRML, PR.

Commons required: APØLLØ, NUTRØN, GØMLØC, ØRGI, PAREM.

Variables required:

X, Y, Z, NMED, U, V, W, NREG} from NUTRØN common, see page

IBLZN - value of IR from last track or from LOOKZ.

ETATH - distance to be traveled (in cm) if the flight remains in the starting medium.

MARK - initial value of flag indicating type of trajectory.

TSIG - total cross section of starting point medium.

DIST - present distance from XB(3).

Variables changed:

X, Y, Z - end point of flight.

NALB - albedo flag (= MEDALB or 0).

MARK - flag indicating type of termination of flight,
0 - normal boundary crossing,
1 - flight within one medium,
-1 - particle escaped,
-2 - particle entered internal void.

NMED - medium of end point.

NREG - region of end point.

ETAUSD - actual flight distance (in m.f.p.).

BLZNT - combinatorial geometry region of end point.

ETATH - actual flight distance (in cm).

DISTØ - distance from XB(3) to rext collision site.

Subroutine GPRØB

This subroutine is the executive routine for the generation and storage of secondary gamma rays (or neutrons for an adjoint, coupled problem). The version of GPRØB which is found to be most useful uses GWL as the probability of generating a gamma ray. Thus, a random number is compared with GWL, and if greater, no gamma ray is generated; if less than or equal, then a gamma ray with weight = WATE*PGEN/GWL is stored. This procedure produces gamma rays of varying weights, but the number of gamma rays may be controlled easily. This version is the one distributed to users.

Another version of GPRØB which has been implemented in some cases uses GWL as a desired gamma weight. The probability of generating a gamma ray is determined and the resulting gamma-ray weight, WATEG, is compared with input values of the desired gamma-ray weight, GWL. Russian roulette and splitting are used to produce gamma rays of weight GWL. That is, if the gamma-ray weight is less than the input values, then the gamma ray is killed with probability (GWL-|WATEG|)/GWL and stored with probability (|WATEG|)/GWL. If the gamma-ray weight is greater than the input value, then there are J = WATEG/GWL gamma rays stored with weight GWL with Russian roulette played with the remaining gamma ray of weight WATEG - J*GWL.

Called from: MØRSE.

Subroutines called: GAMGEN, GSTØRE, FLTRNF.

Commons required: Blank, NUTRØN, APØLLØ.

Variables required:

IG	- primary particle energy group.
NMED	- geometry medium.
WATE	- primary particle weight.
GWL	- input weight values for gamma rays.
NREG	- geometry region.
NMTG	- total number of particle groups.
MXREG	- number of regions for which there are weight standards.

Significant internal variables:

WATEG	- gamma-ray weight.
PGEN	- gamma-ray generation probability.

Subroutine GSTØRE (W8G, IGG)

This subroutine checks to see if there is room in the bank, and if so stores the significant variables for the generated gamma ray (or neutron in an adjoint coupled problem). Since the information in NUTRØN common is stored, the current neutron parameters must be saved temporarily and then restored. It is assumed that the gamma ray is emitted uniformly in direction. An option for analyzing the generated gamma ray is provided through the BANKR interface.

Called from: GPRØB.

Subroutines called: GTISØ (U, V, W), STØRNT (NMEM), BANKR (4).

Commons required: NUTRØN, APØLLØ.

Variables required:

- W8G - gamma-ray weight.
- IGG - gamma-ray energy group.
- LØCNSC - location in blank common of cell zero of scattering counter arrays.
- NMEM - last location in bank that has been used.
- NMØST - maximum number of particles allowed in the bank.
- NMTG - total number of energy groups.
- MXREG - maximum region number.
- IG, NREG, WATE, NAME, U,V,W (from NUTRØN common, see page 4.4-10).

Variables changed:

- NMEM - last location in bank that has been used.
- NEWNM - the gamma-ray name.

Significant internal variables:

- U, V, W - direction cosines of gamma ray.

Limitations: Isotropic gamma-ray emission.

Subroutine INPUT

The basic functions of subroutine INPUT* are to be read, from cards, the basic problem description, and to print out this information, to initialize parameters, to perform some initial transformations on basic problem data, and to call other more specialized routines that perform similar initializations. As an example, several group indices must be set differently depending on whether the problem is a neutron only, gamma only, or combined neutron and gamma. If an adjoint problem is being done, many quantities must be stored differently since all values are input as though a forward calculation was being done.

Called from: MØRSE.

Subroutines called: INPUT1 and INPUT2.

---

*This routine has been split into two routines INPUT1 and INPUT2 to allow for a more efficient overlay structure.

Subroutine INPUT1

This routine reads the random walk input and calls routines to read the source spectra and the geometry data.

Called from: INPUT

Subroutines called:

DATE - provided EBCDIC string containing day of week and data.

SØRIN - reads cards E, source spectra and relative importance of source groups, if biasing is desired.

RNDIN - stores initial random number.

RNDØUT - retrieves current random number.

JØMIN - reads geometry data.

Functions called:

ICØMPA (A,B,N)† (library function at Oak Ridge National Laboratory - compares, bit by bit, N bytes of locations A and B; returns zero if A and B are identical).

MØDEL† (library function at Oak Ridge National Laboratory which determines the model of the computer).

Commons required: Blank, GEØMC, BANK, USER, BNKNMC, NUTRØN, APØLLØ, FISBNK, NØRMAL.

Variables Input: (see definitions of variables in common APØLLØ, NUTRØN, USER, pages 4.4-4,4.4-10,4.6-7)
A more detailed listing of input is given in Section 4.3.)

CARD A (20A4)

Title

(Any character other than a blank or alphameric in column one will terminate the job.)

CARD B (10I5, F5.0, 2I5)

NSTRT, NMØST, NITS, NQUIT, NGPQTN, NGPQTG, NMGP, NMTG, NCØLTP, IADJM, AXTIM, MEDIA, MEDALB

CARD C (4I5,5E10.5)

ISØUR, NGPFS, ISBIAS, (unused), WTSTRT, EBØTN, EBØTG, TCUT, VELTH

CARD D (7E10.4)

XSTRT, YSTRT, ZSTRT, AGSTRT, UINP, VINP, WINP

---

†Not used in versions other than IBM-360.

CARDS E1 (7E10.4) (skipped if ISØUR > 0 or ISØUR = 0 and NGPFS = 0)
(read by SØRIN)
FS(I), I = 1, NGPFS

CARDS E2 (7E10.4) (skipped if ISØUR > 0 or ISØUR = 0 and NGPFS = 0)
(skipped if ISBIAS ≤ 0) (read by SØRIN)
BFS(I), I = 1, NGPFS

CARDS F (7E10.4)
ENER(I), I = 1, NMTG

CARD G (2I5,5X,36I1,5X,13I1)
NHISTR, NHISMX, (NBIND(J),J=1,36),(NCØLLS(J),J=1,13)

CARD H (Z12)
RANDØM

CARD I (14I5)
NSPLT, NKILL, NPAST, NØLEAK, IEBIAS, MXREG, MAXGP

CARDS J (6I5,4E10.5)
NGP1, NDG, NGP2, NRG1, NPRG, NRG2, WTHIHJ, WTLØW1, WTAVE1, PATH
(read until NGP1 < 0)

CARDS K (7E10.4) (skipped if IEBIAS ≤ 0)
((EPROB(IG,NREG),IG=1,NMTG),NREG=1,MXREG)

CARD L (14I5)
NSØUR, MFISTP, NKCALC, NØRMF

CARDS M (7E10.4) (skipped if MFISTP ≤ 0)
(FWLO(I),I=1,MXREG)

CARDS N (7E10.4) (skipped if MFISTP ≤ 0)
(FSE(IG,IMED),IG=1,NMTG),IMED=1,MEDIA)

CARDS O (7E10.4) (skipped if NGPQTN or NGPQTG = 0)
((GWLØ(IG,NREG),IG=1, NMGP or NMTG-NMGP), NREG=1, MXREG)
JØMIN called for geometry data

Variables changed:
All in common USER - set for use by analysis routines.
All in common NØRMAL - zeroed.
All in common GEØMC - zeroed.
All in common NUTRØN - filled with junk word (see page 4.4-19).
All in common APØLLØ - except I1, I0, ITIME, NLAST are filled with junk word.

MAXTIM
DFF
NGPQT0
NGPQT1
NGPQT2
NGPQT3
NWPCØL
NCØLPR

for definitions, see common APØLLØ, page 4.4-4, and Fig. 4.3, the diagram of energy group structure.

RANDØM - set to internal number by RNDIN if zero is read in.

MAXGP - set to 1 if 0 is read in.

MXREG - set to 1 if 0 is read in.

MGPREG - MAXGP*MXREG.

LØCWTS
LØCFWL
LØCEPR
LØCNSC
LØCFSN
NGEØM
NLAST
NSIGL
NFISBN

for definitions, see common APØLLØ, page 4.4-4 .

Subroutine INPUT2

This routine calls routines to read cross section data and analysis data. It also does some initializations and calls a routine to set up the neutron bank.

Called from: INPUT

Subroutines called:

- XSEC - reads cross-section data.
- SETNT - sets up neutron bank.
- EXIT - library.
- SCØRIN - user routine for reading analysis data.
- GAMGEN - provides gamma-generation probabilities.
- FISGEN - provides fission-generation probabilities.
- NSIGTA - picks up cross section.

Commons required: Blank, USER, APØLLØ, FISBNK

Variables input: all the cross section and analysis data.

Variables changed:

NLAST, NSIGL, NFISBN } for definitions see common APØLLØ, page 4.4-4.

Function IWEEK (MØNTH, IDAT, IYEAR) - IBM-360 Version*

This routine will look up the date for you if you don't know it and fill in integer values for MØNTH, IDAT, and IYEAR (requested with MØNTH $\leq$ 0). It also returns, as the function value, an integer from 1 to 7 representing the day of the week. If it is given a positive value of MØNTH, it assumes you have given it a month, day of month, and year and will not disturb these but will simply determine the day of the week. If you stump it (by specifying a year before 1901 or after 2099) IWEEK is returned as zero.

Called by: DATE

Routines called:

IDAY - library routine at ORNL; the output is two 4-byte words containing 8 EBCDIC characters representing the number of the month, a hyphen, the day of the month, a hyphen, and the last two digits of the year. That is, on May 30, 1970, the argument for IDAY will return containing the EBCDIC representation of 05-30-70.

Variables required:

MØNTH $\leq$ 0 - flag to calculate MØNTH, IDAT, and IYEAR.

> 0 - flag to leave arguments alone.

Variables modified:

MØNTH - integer representing month.

IDAT - integer representing day of month.

IYEAR - integer representing year.

IWEEK - integer representing day of week.

*This routine is a dummy routine in the CDC-6600 and UNIVAC-1108 versions.

Subroutine MSØUR

MSØUR is the executive routine for the generation and storage of the source parameters at the starting of each batch. The source parameters may be read into INPUT on cards, generated by subroutine SØURCE or obtained from the fission bank for a multiplying system. For either type of problem the calculations by subroutine SØURCE override the fission bank input or the values read from cards. If the direction cosines are all input as zero, an isotropic source direction is generated. The group number obtained from the fission bank is the group causing fission and may be used in the selection of the source group for the fission neutrons. FSE in blank common contains the group distribution for each medium.

Called from: MØRSE

Subroutines called: FSØLR, GETNT, SØURCE, GTISØ, STØRNT, BANKR(1), LØØKZ

Commons required: NUTRØN, FISBNK, APØLLØ, GØMLØC, ØRGI

Variables required:

ITSTR - an index which determines if the source should be obtained from the previous batch fissions (ITSTR ≠ 0) or generated by SØURCE or from input data (ITSTR = 0).

ISØUR - an index which determines the options for the energy distribution of the source. If ISØUR > 0 the source energies are all generated in energy group ISØUR. If ISØUR < 0, or if ISØUR = 0 and NGPFS ≠ 0, subroutine INPUT1 calls SØRIN and the energy is selected by SØURCE.

NMEM - the number of particles to be generated for the batch, = NSTART for non-fissioning systems and NFISH for multiplying systems.

XSTRT, YSTRT, ZSTRT
WTSTRT, AGSTRT
UINP, VINP, ZINP
} starting parameters input from cards, from common APØLLØ, see page 4.4-4 .

Variables changed:

UØLD, VØLD, WØLD, ETATH, XØLD, YØLD, ZØLD, IBLZØ, ETA, IGØ, MEDØLD, ØLDAGE - previous collision parameters are zeroed for the source.

ØLDWT - previous collision weight set equal to WTSTRT.

X, Y, Z, WATE, AGE, NAMEX,
IBLZN, NREG, NMED, NAME,
U, V, W, IG — parameters set for each particle generated, put in NUTRØN common, see page 4.4-10.

NPSCL(1) - counter for number of sources.

NEWNM - set to name of last particle generated.

FTØTL
FWATE
NFISH — zeroed for the next batch.

Subroutine NXTCØL

This subroutine is called by the main program to determine the spatial coordinates, the block and zone number, particle's age, and non-absorption probability at the next collision site and at every boundary crossing encountered along the way. The total number of boundary crossings is recorded as is the number of escapes. If a particle escapes, its weight is set equal to zero and the history will be terminated by the main program.

Called from: MØRSE

Subroutines called: GETETA, NSIGTA, GØMST, BANKR(7), BANKR(8)

Commons required: Blank, NUTRØN, APØLLØ

Variables required:

- AGE - chronological age of the particle at the previous collision site.
- IBLZN - an integer specifying the zone number at the previous collision site.
- NMED - the medium number at the previous collision site.
- XØLD, YØLD, ZØLD - spatial coordinates at the previous collision site.
- UØLD, VØLD, WØLD - the particle's precollision direction cosines.
- TSIG - total cross section.

Variables changed:

- AGE - chronological age at new collision site.
- IBLZN - an integer containing the zone number at the new collision site.
- NMED - end-of-flight medium.
- NPSCL(7) - total number of boundary crossings.
- NPSCL(8) - number of escapes.
- X, Y, Z - end-of-flight spatial coordinates.
- WATE - weight of particle undergoing flight to the new collision site.

Significant internal variables:

- MARK - an index which identifies the type of event at (X,Y,Z); MARK = 0, normal boundary crossing; MARK = 1, flight ended within the medium; MARK = -1, particle escaped; MARK = -2, particle entered an interior void.

ETA - mean-free paths of flight remaining after a boundary crossing.

ETATH - total distance that a particle would travel if the medium at the starting point was extended indefinitely.

ETAUSD - mean-free paths of flight consumed while traversing a given medium.

Subroutine ØUTPT (KEY)

This routine controls the calculation and output of the average values of the source parameters (beginning of the batch, KEY = 1) and the collision counters at the end of each batch (KEY = 2). At the end of the run (KEY = 3), results for the number of scatterings, the ways in which the particles were terminated, and the counters for splitting and Russian roulette are printed.

For k calculations, the estimate of k at the end of each batch is output, with the final value of k and its standard deviation output at the end of the run.

In addition, the c.p.u. time used is output for each batch.

Called from: MØRSE

Subroutines called: TIMER, GETNT, ØUTPT2

Commons required: Blank, NUTRØN, APØLLØ, FISBNK

Variables required: (nearly all variables from NUTRØN common, see page 4.4-10 for definition).

NITS, ITERS, NMEM

NPSCL(I)

NØRMF, NFISH, NKCALC, NSPLT, NKILL (from common APØLLØ, see page 4.4-4).

Significant internal variables:

FNKFW - a running count of the total number of particles starting.

SWATE - the sum of the source particle weights.

FKSUM - running sum of the k values weighted by the number of particles starting the batch.

VARK - running sum of the square of the k values weighted by the number of particles starting the batch.

NITSK - number of batches used for k calculation.

Subroutine ØUTPT2 (NI, WNI, MAXGP, MXREG, IØ)

This subroutine is used to output the number (NI) and weight (WNI) counters indicating the results of Russian roulette, splitting, and scatterings for the complete problem. The output arrays depend on region and energy group.

Called from: ØUTPT

Variables required:

NI	- the two-dimensional array to be output.
WNI	- the two-dimensional array to be output.
MAXGP	- the largest group for which Russian roulette and splitting were considered.
MXREG	- the number of regions for weight standards.
IØ	- the logical output tape number.

Subroutine SØRIN (DFF, NGPFS)

The source energy spectrum (in group form) and, if needed for biasing, the relative importance of source groups, are input and transformed to cumulative distribution functions (c.d.f.) by this routine. (Note that the biased spectrum is not input but rather calculated by SØRIN.) Forward and adjoint cases are handled automatically. If an adjoint problem is being done, the c.d.f.'s start at 1.0 and decrease with group so they will be in the correct order after INPUT2 reverses the arrays. NGPFS values of the natural spectrum (referred to as the array FS) and, if requested, the relative importance (referred to as the array BFS) are input into blank common. After FS is input, the summations DDF over groups 1 to NGPFS, and DFF over all groups actually being used up to NGPFS are formed. DDF is replaced by (DFF/DDF)*WTSTRT for use, in SØURCE, as a weight correction when less than NGPFS groups are being used in the problem. DFF is transferred to common USER for use by the analysis as a normalization in adjoint problems. It should be noted that the array FS, as input, is treated as fractions of particles to be emitted in the natural distributions, but, for the adjoint case, should consist of averages over the group width, not integrals.

Called from: INPUT1

Functions used: ABS, MAXO (library)

Commons required: APØLLØ

Variables required:

- NGPFS - number of values of FS (and BFS) to be read.
- NMTG - total number of groups in cross sections.
- NGPQTN - number of neutron groups.
- NGPQTG - number of gamma-ray groups.
- NMGP - number of primary particle groups in cross sections.
- WTSTRT - starting particle weight, as input.
- IADJM - positive for adjoint problem, ≤ 0 for forward.
- ISBIAS - source bias switch, biasing used if > 0.

Variables input:

FS(I), I=1, NGPFS, and
if ISBIAS > 0,
BFS(I), I=1, NGPFS

format (7E10.4)

Variables changed:

DFF - summation of FS over groups being used in problem.

DDF - ratio of DFF to summation of FS over NGPFS groups, times starting weight.

Significant internal variables:

NGPQT - set to NGPQTN if neutron only or combined problem, set to NGPQTG if gamma only problem.

NG1 - set to the largest of NGPFS, NGPQTN, NGPQTG for single particle problem, set to NMGP+1 for combined problem.

NG2 - set to NMGP+NGPQTG for combined problem, irrelevant for single particle type problem.

Subroutine SØURCE (IG, U, V, W, X, Y, Z, WATE, MED, AG, ISØUR, ITSTR, NGPQT3, DDF, ISBIAS, NMTG)

This subroutine determines the initial parameters for all primary particles. If the variables which are input to MØRSE are not altered by SØURCE then those input parameters are used for every particle. If a fission problem is being considered, the particle group at the time SØURCE is called is the group causing the fission event and the source energy group for the new particle must be reset. The version of SØURCE discussed here merely selects from an input energy spectrum. An option to select from a biased energy distribution is provided. The weight correction for selecting from the modified distribution is given by the ratio of the natural probability to the biased probability at the selected energy group.

Called from: MSØUR

Commons required: Blank

Variables required:

- ISØUR - a switch which determines the type of source - see INPUT1.
- ITSTR - a switch which indicates whether fission is an original source particle or a daughter.
- NGPQT3 - total number of groups over which the problem is defined.
- DDF - starting weight corrected for source being defined over different number of groups than actually being used in the problem.
- ISBIAS - switch indicating if biased sampling is used for source energy.
- NMTG - total number of groups.

Variables changed:

- WATE - particle source weight.
- IG - particle energy group.

Significant internal variables:

- NWT - location of group zero source probability (either biased or unbiased).

Limitations: This version only selects an energy group. If a user requires selection of other parameters, he must supply an appropriate SØURCE routine.

Subroutine TESTW

TESTW is called after a particle is withdrawn from the bank and then after each collision. A test is first performed to determine if the Russian roulette and splitting options have been specified. Then a comparison of the particle's weight is made with the Russian roulette weight standard WTLØR to determine if the particle will experience Russian roulette. If the particle is killed, its weight is set equal to zero, and if it survives it assumes a new weight, WTAVE, which is designated by the user.

If Russian roulette is not performed, a comparison of the particle's weight is made with the splitting weight standard WTHIR to determine if the particle should be split. If the particle is split, each of the two particles will assume a weight which is half that of the original particle. One of the pair is given a name not in current use, and then placed in the bank. The splitting process is repeated on the remaining particle until the particle's weight falls below the splitting standard WTHIR.

Called from: MØRSE

Subroutines called: BANKR(11)†, BANKR(12)†, STØRNT, BANKR(2)†.

Commons required: Blank, NUTRØN, APØLLØ

Variables input:

IG, MAXGP, NKILL, NSPLT, NMØST, NMEM, NEWNM } from common APØLLØ, see page 4.4-4 .

WTHIR - weight standard for splitting.

WTLØR - weight standard for Russian roulette.

WTAVE - weight assigned to particle which survives Russian roulette.

Variables changed:

WATE - the weight of the particle after splitting or Russian roulette and just before its next collision.

NMEM - the new number of particles in the bank.

NEWNM - the names of the daughter particles created by splitting.

---

†The CALL statements for these routines are comment cards; the user who desires to use them must change this so the statements are executable.

Subroutine TIMER (L,A) IBM-360 Version*

Upon entry to this routine, L is an index having values of -2, -1, 0, or 1, which specify one of the following options:

L	Option
-2	Initialize local and global clocks.
-1	Read global clock.
0	Read and reset local clock.
1	Read local clock.

For all except L = -2, the appropriate clock reading is converted to an EBCDIC string of up to 39 bytes. If the number of hours is zero, only minutes and seconds are provided. If both the number of hours and minutes are zero, only the number of seconds is provided. If all three are zero, the string is 'LESS THAN ONE SECOND'. The number of 4-byte words necessary to contain the string is returned in L.

Typical Usage of IBM-360 Version:

```
     DIMENSIØN ARRAY  (10)
     CALL TIMER  (-2, ARRAY)
     DØ 1 I = 1, 10
     LENGTH = 0
     CALL TIMER  (LENGTH, ARRAY)
   1 PRINT 2, I, (ARRAY(J), J = 1, LENGTH)
   2 FØRMAT ('TIME REQUIRED FØR THE' ,I4,' TIME THRU THIS LØØP WAS ',10A4)
     LENGTH = -1
     CALL TIMER  (LENGTH, ARRAY)
     PRINT 3, (ARRAY(I), I = 1, LENGTH)
   3 FØRMAT ('TØTAL TIME FØR THIS CALC. WAS ',10A4)
```

Called by: MØRSE, ØUTPT

Routines called:

ICLØCK - library function at Oak Ridge National Laboratory; returns reading of computer timer (c.p.u. time) in hundredths of seconds.

*The CDC-6600 and UNIVAC-1108 versions are described on page 4.4-59 of this section.

INTBCD - library subroutine at ORNL; converts a 4-byte integer to an EBCDIC string; also returns the length of the string.

INSERT - library subroutine at ORNL; inserts a string of given length at a specified point in another string.

Variables required:

L - see above.

Variables modified:

A - see above.

Subroutine TIMER - UNIVAC-1108 and CDC-6600 Version

This routine returns time in microseconds. The UNIVAC-1108 version calls a subroutine CPUTIM which uses a function routine TICKER. CPUTIM assumes that the time is returned in units of microseconds. The CDC-6600 version calls CPUTIM which in this case uses a function routine SECØND.

Called from: INPUT1

Subroutines called: CPUTIM which calls TICKER (1108 version) or SECØND (6600 version)

4.4.4. Energy Indexing in MØRSE

For most problems the user does not need to worry about the energy-indexing scheme used in the MØRSE Monte Carlo code. The energies corresponding to the group boundaries in the forward group structure are input and the code takes care of the rest. The analysis package performs the proper bookkeeping for labeling. However, in using the energies in other user routines, some clarification of the indexing scheme for several types of problems is advantageous. This clarification can perhaps best be made through the use of examples. Consider a few-group coupled neutron-gamma-ray problem with the following description:

Group	Energy (eV)	with NMGP = 3,	NMTG = 5
1	15(+6)	NGPQTN = 3,	NGPQTG = 2
2	5(+6)		
3	1(+6)		
EBØTN	3(+3)		
4	10(+6)		
5	2(+6)		
EBØTG	4(+5)		

There are several options which might be considered; namely (1) a forward coupled problem, (2) a forward neutron-only problem, (3) a forward gamma-ray-only problem, (4) an adjoint coupled problem, (5) an adjoint neutron-only problem, and (6) an adjoint gamma-ray-only problem. Table 4.13 gives the energies as they are indexed during a MØRSE run. Table 4.14 gives the values of variables NQT1, NQT2, and NQT3 as they appear in USER common for the same six problems.

Table 4.13. Numerical Examples for Various Options Corresponding Energies for Each Case

Group Index	N + γ Forward	N Forward	γ Forward	N + γ Adjoint	N Adjoint	γ Adjoint
1	15(+6)	15(+6)	10(+6)	2(+6)	1.0(+6)	2(+6)
2	5(+6)	5(+6)	2(+6)	10(+6)	5(+6)	10(+6)
3	1(+6)	1(+6)	--	1(+6)	15(+6)	--
4	10(+6)	--	--	5(+6)	--	--
5	2(+6)	--	--	15(+6)	--	--
EBØTN	3(+3)	3(+3)	--	3(+3)	3(+3)	--
EBØTG	4(+5)	--	4(+5)	4(+5)	--	4(+5)
NMGP	3	3	2	3	3	2
NMTG	5	3	2	5	3	2
NGPQTN*	3 2†	3 2	0 0	3 2	3 2	0 0
NGPQTG*	2 1	0 0	2 1	2 1	0 0	2 1
NGPQT1	3 2	3 2	2 1	2 2	0 0	2 2
NGPQT2	3 3	3 3	0 0	2 3	0 1	0 1
NGPQT3	5 4	3 2	2 1	5 5	3 3	2 2

*Note: A value of NGPQTN = 0 signals a gamma-ray-only problem in the random walk module. A value of NGP = 0 signals a gamma-ray-only problem in the cross-section module and NGPQTG = 0 indicates a neutron-only problem in MØRSE.

†Values are given for two cases with different values of NGPQTN and NGPQTG.

Table 4.14. Values of NGPQT1, NGPQT2, and NGPQT3 for Several Cases

Cases	NGPQT1=	NGPQT2=	NGPQT3=
Forward			
Combined	NGPQTN	NMGP	NMGP + NGPQTG
N Only	NGPQTN	NMGP	NGPQTN = NGPQT1
γ Only	NGPQTG	NMGP*	NGPQTG = NGPQT1
Adjoint			
Combined	NMTG - NMGP	NMTG - NGPQTN	NMTG
N Only	NMTG - NMGP	NMTG - NGPQTN	NMTG
γ Only	NMTG - NMGP*	NMTG - NGPQTG	NMTG

*NMGP is 0 here, but was > 0 on INPUT.

4.5. MULTIGROUP CROSS-SECTION MODULE

4.5.1. Introduction

The function of this module in the multigroup Monte Carlo code is to read ANISN-type[2] cross sections for media or elements, mix several elements together to obtain media cross sections, determine group-to-group transfer probabilities and determine the probabilities and angles of scattering for each group-to-group transfer. All variables are flexibly dimensioned and are part of blank common.

Various types of cross sections can be processed by the cross-section module. Neutron only, gamma ray only, neutron gamma-ray coupled, or gamma rays from coupled neutron gamma-ray cross sections can be processed for either a forward or an adjoint problem with or without fission. The Legendre coefficients are stored if a next-event estimator is to be used.

The cross sections are read for one coefficient and one element into a buffer area. Then these cross sections are decomposed into total, fission, and downscatter matrix and stored in temporary arrays so that they may be mixed to form media cross sections. The total and fission cross sections are stored only once for an element, but the downscatter matrix is stored for each coefficient. The cross sections are transposed as stored if an adjoint problem is being solved.

After all cross sections are stored, the contribution of each element to the cross section for the media is determined. Also at this time the sum of the downscatter vector for each group is determined for the future calculation of the nonabsorption probability; the gamma-production cross section is also determined by summing the transfers to the gamma groups. After the cross sections for the medium have been determined, the nonabsorption probability, fission probability, and gamma-production probabilities are formed by dividing by the total cross section. The downscatter matrix is converted to a probability table by dividing by the scattering cross section.

The Legendre coefficients for each group-to-group transfer are converted to angles and probabilities of scattering at those angles by the use of a generalized Gaussian quadrature using the angular distribution

as a weight function. That is,

$$\int_{-1}^{+1} f(\mu)\omega(\mu)d = \sum_{i=1}^{n} f(\mu_i)\omega_i,$$

where

$f(\mu)$ is any polynomial of order 2n-1 or less,

$\omega(\mu)$ is the angular distribution for μ, the cosine of the scattering angle,

μ_i is a set of discrete cosines,

ω_i is the probability of the corresponding cosine.

Thus, a set of μ_i's and ω_i's that satisfy the equation must be found. To do this, a set of polynomials, Q_i, which is orthogonal with respect to the angular distribution, is defined such that

$$\int_{-1}^{+1} Q_i(\mu)Q_j(\mu)\omega(\mu)d\mu = \delta_{ij}N_i,$$

where N_i is a normalization constant.

The moments of the angular distribution M_i, i=1, 2n-1, determine the orthogonal polynomials, Q_i, i=1, n. The desired cosines, μ_i, are given by the roots of Q_n,

$$Q_n(\mu_i) = 0,$$

and the corresponding probabilities are

$$\omega_i = \left(\sum_{k=1}^{n-1} \frac{Q_k^2(\mu_i)}{N_i} \right)^{-1}$$

In the process of deriving the orthogonal polynomials, some restrictions on the moments of the angular distribution are obtained. These restrictions arise if both the original distribution and the derived point distribution are to be everywhere non-negative. The restrictions are:

1) $N_i > 0$ for i=1, n.
   This restriction may be written in terms of the determinant of the moments:

$$\begin{vmatrix} 1 & M_1 & M_2 & \cdots & M_i \\ M_1 & M_2 & M_3 & \cdots & M_{i+1} \\ \cdot & & & & \\ \cdot & & & & \\ \cdot & & & & \\ \cdot & & & & \\ M_i & M_{i+1} & M_{i+2} & \cdots & M_{2i} \end{vmatrix} > 0$$

2) The roots of $Q_i(\mu)$ must all lie in the interval

$$-1 \leq \mu_i \leq 1.$$

It must be emphasized that the restriction arising from the original distribution being everywhere positive (or zero) does not restrict the truncated expansion of the distribution to be everywhere positive. That is, moments from a truncated distribution that is not necessarily everywhere positive are used to derive a discrete distribution with positive probabilities.

Other characteristics of this representation are that the information is compact, the angles are clustered where the angular distribution is peaked, and because of the restrictions, cross sections that have blunders in them are rejected because they produce angles outside the range of -1 to +1.

An option on input makes it possible to write a tape containing the processed cross sections and all variables from Common LØCSIG needed during the random walk process. In starting a new case, the normal cross-section input, except for the cross sections and the mixing cards, is required. Both the information from input and from tape are printed for compatibility checks. Note that if an adjoint problem is being solved, the input information and the information for tape will not be identical.

One of the more important options provided for in this package is the ability to treat upscattering. Thus, multigroup cross sections with many thermal groups may be utilized. (In a crude way photoneutron production may be treated as an upscatter process from a gamma-ray group to a neutron group.) Also, the capability of using cross sections with partial downscatter is present.

The downscatter array made it possible to have a feature that is helpful in reducing cross-section storage requirements for neutron problems involving materials other than hydrogen when a cross-section tape is used. For non-hydrogeneous media there is very seldom complete downscatter to thermal energy, so a routine was added to search the P_0 array to determine the minimum downscattered energy group for any element and for each group. The zero cross sections for downscatters below this energy group are not stored, and thus the storage requirements are reduced.

Another important feature of this module is the ability to use a point cross-section representation for the total, scattering, and fission cross sections. Either an Ø5R[6] or an Ø6R[7,8] (variable supergroup boundaries) format tape (CØDE7) may be used over a specified energy range. (This report describes the routines to be used with an Ø6R tape.) The energy of a particle is chosen uniformly within an energy group for the purpose of selecting the particle track lengths or calculating a non-absorption probability. For gamma rays or for neutron energies outside the specified range, the total cross section from the multigroup cross sections is used. (Point, total, and scattering cross sections for the adjoint case can also be used.) For some applications, detailed non-absorption probabilities and fission probabilities may also be required. The logic was included in the new module for the cross-section manipulation required for the use of the point cross sections. Up to 16 point cross-section media with up to 100 supergroups are allowed.

One major change in philosophy has been made for the case in which point cross sections are used. The modular framework of MØRSE is broken in this case in that Subroutine GTSCT uses the energies corresponding to the multigroup boundaries in order to set up an array of indices. It is assumed by GTSCT that the energies exist in the first NTG cells of Blank Common.

Figure 4.4 shows the hierarchy of the subroutines in the cross-section module. Table 4.15 gives the definitions of the variables in Common LØCCTG which contains all the variables used to locate cross sections. The location of permanent cross-section information in Blank Common is presented in Table 4.16. Table 4.17 contains the definitions of variables in Common

GTSC1, a common containing information about the point cross-section data. Other details of the cross-section module are given with the description of the various subroutines. A more detailed description of the theory for the generalized Gaussian quadrature is given in Section 4.11.

An executive program XCHEKR[9] may be used in conjunction with this cross-section module independent of MØRSE for the purpose of checking and editing multigroup cross sections. Also, this executive program may be used to generate a processed cross-section tape independent of a MØRSE calculation.

4.5.2 Blank Common Cross Section Storage Requirements

Total permanent storage is 2*NTG + NMED*ISPØRG + 3*NMIX + IX + LEG + NPT, where

NTG = NGP + NGG

ISPØRG = 5*NTG + NGP(1 + NGG) + (2*NSCT + 1)NDSNGP + (2*NSCT + 1)NDSNGG

IX = NELEM*NCØEF if IXTAPE > 0

= 0 otherwise, and

LEG = (NDSNGP + NDSNGG) * NMED*(NCØEF-1) if ISTAT > 0

= 0 otherwise.

NDSNGP = determined by code if IXTAPE > 0 but less than or equal to $\frac{(NDS + NUS + 1)(NDS + NUS)}{2}$

NDSNGG = determined by code if IXTAPE > 0 but less than or equal to $\frac{(NDSG + NUS + 1)(NDSG + NUS)}{2}$

$$NPT = (IGQPT + 1)*NMED + \sum_{L=1}^{NMED} NPT(L)*(NEGPS - 1)*NXPM \text{ if } IØ6RT > 0$$

= 0 otherwise.

Temporary storage, which is storage used only during the mixing, is in addition to the above. The amount of temporary storage is

NELEM*(NTG*(NTS+2) + (NCØEF-1)*(NDSNGP + NDSNGG))

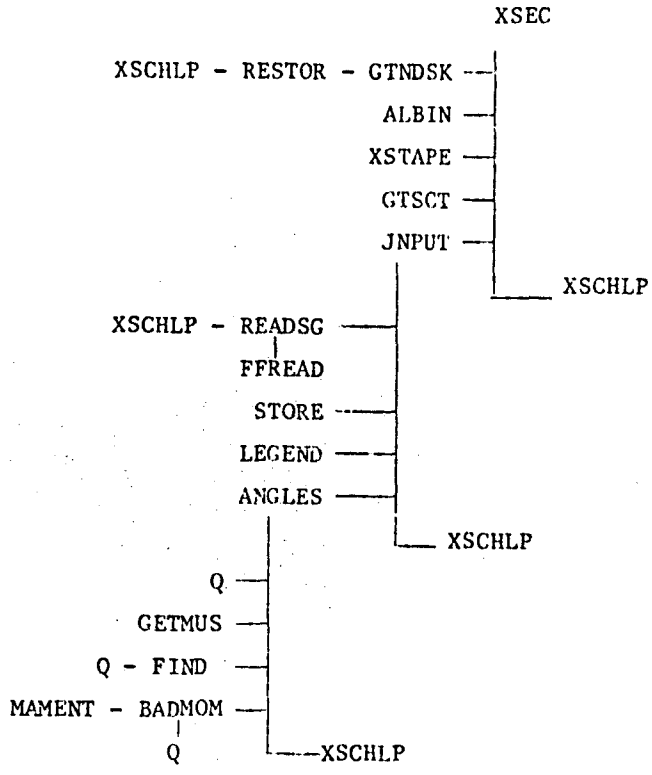
where

NTS = NDS + NDSG + NUS

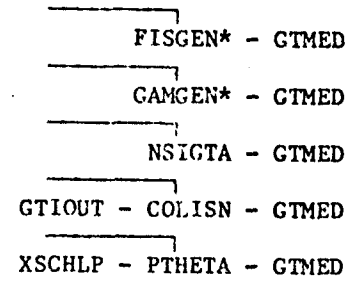
NDSNGP and NDSNGG are as defined above.

Because some overlapping of permanent and temporary storage is allowed when the size of blank common is less than the sum of the two areas, the total storage used will usually be less than the sum.

Routines Called During Problem Initialization



Routines Called During Random Walk



*FISGEN and GAMGEN are also called during MORSE problem initialization.

Fig. 4.4. Hierarchy of Subroutines in the MORSEC Module

Table 4.15. Definitions of Variables in Common LØCSIG

Variable	Definition
ISTART	starting location for the total cross-section vector for the first medium
ISCCØG	starting location for the scattering cross-section vector for the first medium
INABØG	starting location for the non-absorption vector for the first medium
IGABØG	starting location for the gamma-ray production vector for the first medium
IFPØRG	starting location for $\nu\Sigma_f$, the fission neutron production vector for the first medium
IFNGP	starting location for the primary-secondary tranfer probability matrix
IFSPØG	starting location of the primary downscatter probability matrix
IDSGØG	starting location of the secondary downscatter probability matrix
IPRBNG	starting location of the primary scattering angle probability matrix
IPRBGG	starting location of the secondary scattering angle probability matrix
ISCANG	starting location of the primary scattering angle matrix
ISCAGG	starting location of the secondary scattering angle matrix
ISPØRG	size of storage needed for each medium, not including Legendre coefficients

Table 4.15 (Cont'd.)

Variable	Definition
ISPØRT*	starting location for temporary storage of downscatter matrix
INPBUF	starting location for temporary storage of the P_0 table
ISIGØG	starting location for temporary storage of total cross section for element 1
INFPØG	starting location for temporary storage of $\nu\Sigma_f$ for element 1
IABSØG	starting location for temporary storage of downscatter matrix for P_L coefficients (primary groups, element 1)
ITØTSG*	total storage required for temporary storage
NGP	the number of primary groups to be treated
NDS	number of downscatters for NGP (usually equal to NGP)
NGG	number of secondary groups to be treated
NDSG	number of downscatters for NGG (usually equal to NGG)
INGP	number of groups for which cross sections are to be input
INDS	number of downscatters for the INGP groups
NMED	number of media for which cross sections are to be stored - should be same as MEDIA as read on Card B of MØRSE input
NELEM	number of elements for which cross sections are to be read
NMIX	number of elements times density operations to be performed
NCØEF	number of coefficients, including P_0

*If Legendre coefficients are to be restored, then:

INFPØG - redefined by JNPUT as number of locations required for each coefficient (both primary and secondary)

ITØTSG - redefined by JNPUT as total storage required for all coefficients for each medium

ISPØRT - redefined by JNPUT as starting location of P_1 coefficient for primary groups for medium 1

Table 4.15 (Cont'd.)

Variable	Definition
NSCT	number of discrete angles (usually NCØEF/$2_{Integral}$) (=0 for P_0)
NTS	number of downscatters for combined primary and secondary groups (usually equal to NTG)
NTG	total number of groups (primary + secondary) = NGP + NGG
NDSNGP	the number of locations needed for the downscatter matrix for the primary particle
NDSNGG	the number of locations needed for the downscatter matrix for the secondary particles
IADJ	same as IADJM
NME	indicator for stripping gamma rays from a coupled neutron gamma-ray cross-section set - set equal to number of neutrons groups + 1
LØC	same as LØCEPR
INGS	starting location of the indices for starting location of the downscatter vector for each group for primary particles
INSG	starting location of the indices for starting location of the downscatter vector for each group for secondary particles
I1, I0	input and output logical unit numbers
KKK	a running index of the number of cross sections that have already been read in (used in checking the element numbers obtained from tape)
IXTAPE	logical tape number of the multigroup cross-section tape if > 0, or logical tape number of the processed cross-section tape if < 0
IDLL	starting location for element identifiers which determine the element cross sections to be read from tape
ITEML	amount of storage for primary and secondary group downscatters per element
ITEMG	starting location for temporary storage of downscatter matrix for P_L coefficients (secondary groups) for element 1

Table 4.15 (Cont'd.)

Variable	Definition
IRSG	starting location of the mixing parameters
IRDSG	switch to print the cross sections and to test the card sequence as they are read if > 0 (test card sequence only if = 0, and does neither if < 0)
ISTR	switch to print cross sections as they are stored if > 0
IPRIN	switch to print angles and probabilities if > 0
IFMU	switch to print intermediate results of μ's calculation if > 0
IMØM	switch to print moments of angular distribution if > 0
IDTF	switch to signal that input format is DTF-IV format if > 0; otherwise, ANISN format is assumed
ISTAT	flag to restore Legendre coefficients for next-flight estimates if > 0
IPUN	switch to print results of bad Legendre coefficients if > 0
NUS	number of groups of upscatter
NGN	not used presently
IHT	location of Σ_T in the cross-section table, currently set to 3 in XSEC
INUS	starting location for the upscattering vector for the first medium
INUSN	number of impossible first moments found in the cross sections
INGN	starting location for the photoneutron production vector for the first medium (not used at present - same as INUS)
INGNP	starting location for the secondary-primary transfer probability matrix (not used at present - same as INUS)
INNN	starting location for the array of the number of downscatter groups for each primary group
IGGG	starting location for the array of the number of downscatter groups for each secondary group

Table 4.16. Location of Permanent Cross Sections in Blank Common

Location†	Information	Size
IRSG = NLAST	List of Mixing Table	3*NMIX
INGS	Index to Σ_{gg} (Primary)	NGP
INSG	Index to Σ_{gg} (Secondary)	NGG
IDEL	List of Element I.D. Numbers	NELEM*NCØEF if IXTAPE > 0
INNN	Number of Down-scatters for each primary group	NGP
IGGG	Number of Down-scatters for each secondary group	NGG
ISTART		
ISCCØG	Σ_T	NTG
INABØG	Σ_s	NTG
IGABØG	Σ_s/Σ_T	NTG
IFPØRG	Σ_γ/Σ_T	NGP
INUS	$\nu\Sigma_f/\Sigma_T$	NTG
INGN	$\Sigma_{g'\to g}$	NTG
INGNP	Σ_N/Σ_T Not used at present	NGG
IFNGP	$\Sigma_{\gamma\to N}$ Not used at present	NGG*NGP
IFSPØG	$\Sigma_{N\to\gamma}$	NGP*NGG
IDSGØG	$\Sigma^N_{g'\to g}$	NDSNGP
	$\Sigma^\gamma_{g'\to g}$	NDSNGG

†Locations are for index of zero.

Table 4.16 (Cont'd.)

Location†	Information	Size
IPRBNG	$P^N_{g'\to g}$	NDSNGP*NSCT
IPRBGG	$P^\gamma_{g'\to g}$	NDSNGG*NSCT
ISCANG	$A^N_{g'\to g}$	NDSNGP*NSCT
ISCAGG	$A_{g'\to g}$	NDSNGG*NSCT
ISPØRG + ISTART	Repeat for next medium	ISPØRG
If ISTAT > 0 ISPØRT	P_l Coefficient Primary	NDSNGP
	P_l Coefficient Secondary	NDSNGG
	Repeat for P_L Coefficient	INFPØG*(NCØE -2)
	Repeat for next medium	INFPØG*(NCØEF-1)
NLAST or NNIC		
NNIC	Index to point cross sections for each multigroup boundary	(IGQPT+1)*NMED if IØ6RT > 0; otherwise zero
NXSECT(1)	Point cross sections for Medium 1: total	NPT(1)*(NEGPS-1)*NXPM
	scattering	
	.	
	.	
	. $\nu\Sigma_f$	
NXSECT(J)	Point cross sections for Medium J: total	NPT(J)*(NEGPS-1)*NXPM
	scattering	
	$\nu\Sigma_f$	
NLAST		

†Locations are for index of zero.

Table 4.17. Definitions of Variables in Common GTSC1

Variable	Definition
NPT(16)	Number of points per supergroup for each medium. (Only 16 media are allowed.) Redefined at end of GTSCT as the total storage required for each cross section for each medium.
NXSECT(17)	Starting location of point cross sections for each medium.
NDSGP	Total storage required per supergroup for all media.
I1, I0	Input and output logical unit numbers.
IØ6RT	Logical tape number of Ø6R point cross-section tape.
IGQPT	Last MØRSE multigroup for which point cross sections will be used ($\leq$ NMGP)
NEGPS	Number of supergroup boundaries.
ESPD(100)	Energy boundaries (in eV) of the supergroups (only 100 supergroups are allowed).
NNIC	Starting location of the point cross sections.
NXPM	Number of cross sections per medium = 1, if total only = 2, if total + scattering = 3, if total, scattering, and ν*fission
NINC	Index for locating point cross sections. Used as flag in FISGEN to determine if transport process has begun.

4.5.3 Subroutines

Subroutine ALBDØ

This routine is called upon encountering an albedo scattering surface and provides the outgoing neutron parameters for the albedo collision.

The sample routine performs specular reflection at the albedo scattering surface. The requirements of specular reflection may be written as

$$I \cdot N = -R \cdot N,$$

and

$$I \times N = R \times N,$$

where I is the incoming neutron direction vector,

R is the reflected neutron direction vector, and

N is the outward normal to the surface ($I \cdot N < 0$).

Manipulation of the above two equations results in

$$R = I - 2(I \cdot N)N.$$

Called from: MØRSE

Commons required: NUTRØN, NØRMAL

Variables required: U, V, W (from common NUTRØN, see page 4.4-10)

UNØRM, VNØRM, WNØRM - components of unit vector normal to boundary.

Variables changed: U, V, W

Subroutine ANGLES (IG1, JG1, MX)

This is the main executive routine for the generalized Gaussian quadrature. First it calls GETMUS which uses the moments of the angular distribution to determine the recurrence relations which generate the orthogonal polynomials. In so doing GETMUS performs the check for $N_i > 0$, which is one of the requirements on the moments. Next ANGLES calls FIND in an iterative fashion in order to calculate the roots of the orthogonal polynomials. FIND checks the roots to determine if the second restriction on the moments, namely that the roots must lie in the interval (-1, +1), is satisfied. Next ANGLES calculates the weight factors associated with each root in the Gaussian quadrature. Finally the angles and probabilities which have been calculated are rearranged so that they appear in order of decreasing probability. If the given moments do not satisfy the two requirements, then it is not possible to determine as many angles and weights as initially requested. However, ANGLES determines as many as it can from the data given. All cross sections with impossible first Legendre coefficients are found before termination of the problems. NOTE: If 2n+1 moments are given (and all are acceptable), then a discrete distribution with n+1 scattering angles may be determined. If only 2n moments are given, then there is a certain amount of freedom in choosing a 2n+1-st moment to complete the calculation. In these cases ANGLES will compute a value of μ_{n+1} (and hence of M_{2n+1}) which is in the middle of the allowed range for μ_{n+1} and, using this value of μ_{n+1}, complete the calculation of a (n+1)-angle distribution.

Called from: JNPUT

Subroutines called: GETMUS, FIND, Q, EXIT, BADMØM, XSCHLP

Commons required: MEAN, RESULT, MØMENT, LØCSIG

Variables required:

IG1, JG1 — indices of group-to-group transfer being calculated.

NMØM — number of moments given.

XMØMNT(I) = M_i, i=1, NMØM.

IMØM — > 0, print moments,
≤ 0, do not print moments.

MX — medium number.

IPUN ≤ 0, do not print error messages,

> 0, print error messages.

IØ - output unit number.

NSCT - number of scattering angles expected.

Variables changed:

PØINT(I) = X_i = cosine of scattering angle for I=1, NV+1.

WEIGHT(I) = W_i = probability of scattering angle for I=1, NV+1.

NM - number of μ values accepted.

NV - number of σ^2 values accepted.

Significant internal variables:

XMU(I) - μ_i

VAR(I) - σ_i^2

XNØRML(I) - N_i

RØØT(I,J) - Ith root of $Q_J(X)$.

NP = NV+1 = number of angles in discrete distribution.

NACC = NM+NV = number of moments accepted.

Output:

XMØMNT(I) = M_i, I=1, NMØM.

Indices of group, number of moments accepted (only if number accepted is less than number given).

Subroutine BADMØM

In the event that a moment has been rejected because it implied negativity in the angular distribution, BADMØM is called to provide a printout to the user giving the value of the quantity rejected, μ_i or σ_i^2, of the moment rejected, and of the Legendre coefficient which was rejected. In addition, the allowed limits on these quantities are also printed out.

See mathematical description for formulas used.

Called from: ANGLES

Subroutines called: MAMENT

Functions used: Q

Commons required: MØMENT, MEANS, QAL, LØCSIG

Variables required:

N - number of σ^2's accepted.

NN - number of μ's accepted.

(NOTE: N=NN implies μ_{N+1} rejected; N < NN implies σ^2_{N+1} rejected.)

MØMENT(I) - M_i

MU(I) - μ_i

VAR(I) - σ_i^2

NØRM(I) - N_i

QR(I) = $q_i = L_i/N_{i-1}$

A(I,K) - a_{ik}

(NOTE: I = i, but K = k+1.)

Significant internal variables:

NM = N+NN = number of moments accepted.

NBAD = NM+1 = index of moment rejected.

NP1 - N+1

NM1 - N-1

Output:

MUT - μ^{max}

MUB - μ^{min}

VART - $(\sigma^2)^{max}$

VARB - $(\sigma^2)^{min}$

MØMT - M^{max}

MØMB - M^{min}

FT - f^{max}

FB - f^{min}

Subroutine CØLISN (IG, U, V, W, WATE, IMED, NREG)

The subroutine is called at each collision and the incoming group number, direction cosines, and particle weight are converted into post-collision parameters. The outgoing group is selected from the downscatter matrix (the vector corresponding to the incoming group). After determining the outgoing group, the cosine of the angle of scattering is determined from the set of probabilities and angles for the particular group-to-group transfer. The outgoing direction cosines in the laboratory coordinate system are determined from the incoming directions and the angle of scattering and a uniformly selected azimuthal angle. These cosines are normalized to 1.0. The particle's weight is altered by the non-absorption probability in lieu of absorption.

As an importance sampling scheme, the outgoing group probability distribution may be altered and selection of the outgoing group is made from this biased distribution. If this option is chosen, LØCEPR > 0, and subroutine GTIØUT is called.

Called from: MØRSE

Subroutines called: GTMED, GTIØUT, GTISØ, AZIRN

Functions used: FLTRNF, SQRT (library)

Commons required: Blank, LØCSIG

Variables required:

IG	- the precollision energy group.
U,V,W	- the precollision direction cosines.
WATE	- precollision particle weight.
IMED	- geometry medium of collision.
NREG	- geometry region of collision.

(Various indices from common LØCSIG, see page 4.5-7 .)

Variables changed:

IG	- post-collision group.
U,V,W	- post-collision direction cosines.
WATE	- post-collision weight.

Significant internal variables:

PNAB	- non-absorption probability.
IH	- group number = IG for primary particle,
	= IG - NGP for secondary particle.

NADDPG - number of locations between starting location of scattering angle probabilities for primary and secondary particles.

R - random number.

IND - location of biasing parameters for group IG.

NDSK - number of downscatter groups.

FM - cosine of polar angle of scattering.

SINETA, CØSETA - sine and cosine of azimuthal angle of scattering.

Limitations: number of angles is equal to number of probabilities for each group (assumed in use of NADDPG).

Subroutine FFREAD (IN, K, V, NF, N5, N6, IPRTRG)

This routine reads the cross section data in either fixed form or free form.† Unless otherwise specified, it assumes fixed form. To specify free-form place a card containing ① ** in columns 1-3 in front of the data to be read. (A circled number, n, indicates n spaces.) Although this routine will accept all the usual options (R, Z, I, T, S, F, A, E, Q, L, N, M, U, V), the calling routine in MØRSE allows only R and Z options because programs such as FFPUN[14] which punch cross sections use only these two options.

Called from: READSG

Variables required.

N5, N6 - logical unit numbers of standard I/Ø tapes.

IPRTRG - signal indicating whether or not to print the card as read.
≤ 0, do not print,
> 0, print.

IN, K, V - must be dimensioned by 37 in any calling program.

Variables changed:

IN(I) - the number of repeats or zeros in a given field, I, if any; otherwise, zero. IN(37) contains the card number.

K(I) - the type of operation to be performed on the data in field (I); if none, it is blank; otherwise may contain R for repeat or Z for zero.

V(I) - the actual numerical value in field I.

NF - the number of fields on the card (≤ 36).

Significant internal variables:

NY(76) - contains the card image as read from cols. 1-76.

NY(77) - contains the card count from columns 77-80.

IFREE - signal indicating whether data is in fixed or free form.
= 1, fixed form (default),
= 0, free form.

---

†See Ref. 13, Appendix B, pp. 58-65.

Subroutine FIND (L,NF)

This subroutine determines if the roots of $Q_L(x)$, the Lth order orthogonal polynomial, lie within the range (-1,+1). If not, a flag, NF, is set to 1 and the subroutine returns. If the roots lie within the range (-1,+1), then NF = 0, and the subroutine proceeds to calculate the roots. The roots, x_k, k = 1, L, are stored in RØØT(K,L), K = 1, L in labelled common RESULT. The roots are in increasing order RØØT(1,L) < RØØT(2,L) < . . . < RØØT(L,L).

FIND presumes that the roots of $Q_{L-1}(x)$ have already been calculated and stored in RØØT(K,L-1), K = 1, L-1. Thus it is necessary to use FIND in a bootstrapping manner. First RØØT(1,1) = M_1, the root of $Q_1(x)$, is stored. Then one sequentially calls FIND(2,NF), FIND(3,NF), etc. It is also presumed that the roots of $Q_{L-1}(x)$ are in the interval (-1,+1).

FIND uses the property of orthogonal polynomials that the roots of Q_L and Q_{L-1} "interleave". Thus:

1) Q_L has no roots above +1 if Q_{L-1} has no roots above +1 and $Q_L(+1) > 0$. (Remember that $Q_L(+\infty) > 0$.)

2) Q_L has no roots below -1 if $Q_L(-1)$ differs in sign from Q_L (RØØT(1,L-1)) where RØØT(1,L-1) is the lowest root of $Q_{L-1}(x)$.

3) The Kth root and no other root of Q_L lies between the K-1th and the Kth roots of Q_{L-1}.

Once the root has been isolated as being between XLØW = RØØT(K-1,L-1) and XUP = RØØT(K,L-1), it is found by a very simple procedure. The interval (XLØW,XUP) is bisected by XTRY = (XLØW + XUP)/2. Then the subinterval containing the root is determined by the fact that the sign of Q_L must change in passing over the root. Thus the root lies in (XLØW,XTRY) if sign [Q_L(XLØW)] ≠ sign [Q_L(XTRY)] and it lies in (XTRY,XUP) otherwise. XTRY replaces the appropriate limit, XUP or XLØW, and the process is repeated. Each iteration reduces the size of the boundary interval by 2, or, in other words, increases the accuracy to which the root is known by one binary bit. Obviously, after as many iterations as the computer word has bits, XTRY will be as close to the root as can be calculated by the computer.

Called from: ANGLES

Subroutines called: Q

Commons required: RESULT, MACSIG

Variables required:

L - the order of the polynomial whose roots are desired.

RØØT(K,L-1), K=1,L-1 - the roots of $Q_{L-1}(x)$ in increasing order.

IPUN $\begin{cases} \leq 0, & \text{do not print error message,} \\ > 0, & \text{print error message.} \end{cases}$

Variables changed:

NF $\begin{cases} = 0, & \text{the roots of } Q_L(x) \text{ lie in the interval } (-1,+1), \\ = 1, & \text{the roots of } Q_L(x) \text{ do not lie in the interval } (-1,+1). \end{cases}$

If NF = 0

RØØT(K,L), K=1, L - the roots of $Q_L(x)$, in increasing order.

Significant internal variables:

VALUE(K) = Q_L(RØØT(K)), K = 1, L-1.

LM1 = L - 1.

NSP - number of iterations taken in root-finding procedure.

Limitations: L $\leq$ 14.

Subroutine FISGEN (IG, IMED, PNUF)

This subroutine looks up the value of $\nu\Sigma_f/\Sigma_T$ for the current neutron energy and geometry medium.

Called from: INPUT2

Subroutines called: GTMED

Commons required: Blank, LØCSIG

Variables required: ISPØRG, IFPØRG, IG, MED } (from common LØCSIG, page 4.5-7)

Variables changed: PNUF

Subroutine GAMGEN (IG, IMED, PGEN, IGG)

This subroutine provides the function of determining the energy of the secondary particle to be generated and its probability of generation. For a forward neutron gamma-ray problem, a neutron of energy IG upon suffering a collision in medium IMED may generate a secondary gamma ray of energy IGG. For an adjoint gamma-ray neutron problem, a gamma ray of energy IG generates a neutron of energy IGG. Only secondary source gamma-ray groups of higher energy than the lowest gamma-ray group of interest may be chosen.

Called from: GPRØB, INPUT2

Subroutines called: GTMED

Functions used: FLTRNF

Commons required: Blank, LØCSIG, USER

Variables required: ISPØRG, IFNGP, NGG, IGABØG, IADJ (from common LØCSIG, see page 4.5-7).

IG - incoming energy group.

IMED - medium of collision site as provided by the geometry module.

NGPQT1, NGPQT2, NGPQT3, NGPQTG (from common USER, see page 4.6-7).

Variables changed: PGEN, IGG

Subroutine GETMUS

This subroutine calculates the quantities μ_i and σ_i^2 used in the recurrence relation for the orthogonal polynomials, $Q_i(x)$. It uses as input the moments, M_i, of the distribution f(x). GETMUS also checks to determine if $\sigma_i^2 > 0$. If not, a flag is set to indicate this.

Let us assume that NMØM moments are given initially. Then NMØM = NM + NV where NV = NMØM/2 is the number of σ_i^2 quantities to be calculated and NM is the number of μ_i quantities to be calculated. NM = NV or NM = NV + 1, depending on whether NMØM is even or odd. GETMUS calculates μ_i = 1, NM and σ_i^2, i = 1, NV. This is sufficient to determine $Q_i(x)$ for i = 0, NM. If it turns out that some value of σ_i^2 is not positive, say $\sigma_p^2 \leq 0$ (this will happen when $N_p \leq 0$, a violation of the "non-negativity" condition on f(x)), then the calculation is terminated, a flag is set, and GETMUS returns with NV = p - 1 and NM = p.

The relevant equations are as follows:
The orthogonal polynomials are written

$$Q_i(x) = \sum_{k=0}^{i} a_{ik} x^k \text{ with } a_{ii} = 1$$

$$= (x - \mu_i) Q_{i-1}(x) - \sigma_{i-1}^2 Q_{i-2}(x).$$

This leads to

$$a_{ik} = a_{i-1,k-1} - \mu_i a_{i-1,k} - \sigma_{i-1}^2 a_{i-2,k}.$$

If we define

$$N_i = \sum_{k=0}^{i} a_{ik} M_{i+k},$$

$$L_i = \sum_{k=0}^{i-1} a_{i-1,k} M_{k+i}, \text{ and}$$

$$q_i = L_i / N_{i-1}.$$

Then we have

$$\mu_i = q_i - q_{i-1}, \text{ and}$$

$$\sigma_i^2 = N_i / N_{i-1}.$$

The calculation proceeds as follows:

Step 1: initial values for quantities for i = 1 and 2 are set up from explicit formulas from the moments.

Step 2: set i = 3.

Step 3: calculate L_i from moments and coefficients for i - 1.

Step 4: calculate q_i from L_i and N_{i-1}.

Step 5: $\mu_i = q_i - q_{i-1}$.

Step 6: calculate a_{ik}, k = 0, i from μ_i, σ^2_{i-1}, and $a_{i-1,k}$.

Step 7: calculate N_i from moments and $a_{i,k}$'s.

Step 8: $\sigma^2_i = N_i/N_{i-1}$.

Step 9: Test σ^2_i. If $\sigma^2_i \leq 0$, terminate the calculation with n = i-1 and set error flag.

Step 10: i = i+1, return to step 3.

If NMØM is even, the calculation terminates after step 9 when i = NMØM/2.

If NMØM is odd, the calculation terminates at step 5 when i = (NMØM+1/2).

Called from: ANGLES

Commons required: MØMENT, MEANS, QAL, LØCSIC

Variables required: MØMENT(k) = M_k, k = 1, NMØM (type real).

IFMU { ≠ 0, print out all the quantities calculated by GETMUS,
= 0, do not print out data except in case of error, ($\sigma^2_k \leq 0$).

Variables changed:

NV - the number of σ^2's calculated.

NM - the number of μ's calculated.

MU(I) = μ_i, i = 1, NM (type real).

SIG(I) = σ^2_i, i = 1, NV.

NØRM(I) = N_i, i = 1, NV (type real).

Also calculated and put in labelled common QAL, although they are not used elsewhere in the program,

Q(I) = q_i, i = 1, NM.

A(I,K) = $\mu_{i,k-1}$, i = 1, NV; K = 1, i+1

L(I) = L_i, i = 1, NM.

Limitations: NMOM ≤ 27.

Subroutine GTIØUT (IS, J, NREG, NDSK, IG, WATE, IND)

This subroutine is called when the selection of the group-to-group transfer is to be biased. Thus, the natural probabilities of scatter from group I to group J, P(I→J), is to be altered by an importance function V(J). Selection of the outgoing group L is made from P(I→J)V(J) with an associated weight correction of N/[V(L)] where

$$N = \sum_{J=1}^{NDSK} V(J)P(I \rightarrow J).$$

Called from: CØLISN

Functions used: FLTRNF

Commons required: Blank

Variables required:

IS	- one less than index for within-group scattering,
NREG	- geometrical region of the collision,
NDSK	- number of possible downscatter groups.
IG	- incoming energy group.
WATE	- incoming particle weight.
IND	- index for the location of importance of within-group scattering.

Variables changed:

J	- the number of downscattering groups.
WATE	- modified to correct for the biasing.

Significant internal variables:

SBSIG is the normalization N of the biased distribution.

Subroutine GTMED (MDGEØM, MDXSEC)

The standard version of this routine is actually a dummy routine because it assumes that the geometry media numbers and the cross section media numbers are identical.† If this is not the case, the user must supply a routine which specifies the relationship between the two.

Called from: CØLISN, FISGEN, GAMGEN, NSIGTA, MØRSE, PTHETA

Variables required:

MDGEØM - the media number of the geometry media.

Variables changed:

MDXSEC - the cross section media number corresponding to the geometry media MDGEØM.

†Note: The need for this routine arose because the 05R type geometries detected a boundary crossing only if there was a different media on each side of the boundary. However, for a homogeneous problem, the transport needed only one cross section media to be stored. Therefore, this routine was written to allow one to equate the cross sections for two geometric media. Since the advent of the combinatorial geometry package (CG), the need for such a routine is virtually non-existent because CG uses a different technique for determining boundary crossings. With just a minimum of effort the user of MORSE-CG can eliminate the need for this routine by equivalencing MDGEOM & MDXSEC and removing the calls to this routine.

Subroutine GTNDSK

This routine determines the maximum number of non-zero downscatters for each group for all elements. The neutron part of the P_0 table is searched for each element with the downscatter for each group determined by the maximum non-zero group-to-group transfer cross section. The maximum value found for any element is used for all media. After searching the P_0 table, the input cross-section tape is rewound and read again in the normal cross-section input. If hydrogen is present, complete downscatter will be needed and no advantage is gained in using this routine; in this case, a dummy routine may be substituted.

Called from: XSEC if IXTAPE > 0

Subroutine called: RESTØR

Commons required: Blank, LØCSIG

Variables required: All variables in Common LØCSIG

Variables changed: Downscatter index array in Blank Common

Significant internal variables:

INP - starting location in Blank Common for temporarily storing P_0 matrix.

IST - starting location in Blank Common of restored cross section.

ISP - last location in Blank Common used.

N1 - first group to be considered.

N2 - last group to be considered.

Subroutine GTSCT (NLAST, NLFT)

The function of Subroutine GTSCT is to read an Ø6R[7,8] point cross-section tape, to store the cross sections in Blank Common, and to set up indices which give the location in Blank Common of the cross sections corresponding to the energy boundaries in the multigroup structure. This routine has been written to process the data from an Ø6R tape with variable supergroup boundaries; the energy limits of the boundaries are read from the tape. (To process an Ø5R[6] tape, the tape read statement would need to be modified and the supergroup energy boundaries provided separately.)

A variable number of points per supergroup may be used for each media for which point cross sections are provided; however, if total scattering and ν*fission cross sections are used, then the number of points for these individual cross sections must be the same within a medium. In order to reduce the time required to choose a point cross section for a given energy, indices corresponding to the multigroup energy boundaries are calculated and then an index is chosen within a multigroup. In making this index calculation, it is assumed that the energy array is the first array in Blank Common. Then in Subroutine NSIGTA an index is chosen uniformly between the indices for the multigroup boundaries. This is equivalent to assuming a uniform flux within a group. Other schemes for picking an energy within a group can be easily incorporated.

Called from: XSEC

Commons required: Blank, GTSC1, USER

Variables required:

NLAST - the last cell used in Blank Common before GTSCT was called.

NLFT - the last cell in Blank Common that can be used.

IØ6RT - logical unit for an Ø6R point cross-section tape.

IGQPT - the last multigroup for which point cross sections will be used.

There is one card read by GTSCT which contains NXPM, the number of different types of cross sections per medium.

NXPM = 1, total cross section only;
= 2, total + scattering;
= 3, total, scattering, and $\nu*\Sigma_f$.

Additional input information is read from the first record on tape. These variables are:

ETØPX - the highest energy for which there are point cross sections.

EBØTX - the lowest energy for which there are point cross sections.

NSIGX - the number of cross sections on tape.

IDLM(L) - the element identification number for the Lth cross section.

IDSG(L) - the element type for the Lth cross section.

NPT(L) - the number of points per supergroup for the Lth cross section.

NEGPS - the number of supergroup boundaries.

ESPD(I) - the energy corresponding to the upper boundary of the Ith supergroup.

The point cross sections are read into a temporary storage area and then reordered and stored permanently (see Table 4.16 for storage arrangement). Variables changed:

NPT(L) - redefined as the total storage required for each cross section for the Lth medium ($L \leq 16$).

NLAST - the last cell of permanent storage used.

NXSECT(L) - starting location in Blank Common of point cross sections for the Lth medium ($L \leq 16$).

Subroutine JNPUT

This subroutine is the executive routine for processing the cross sections from the ANISN[2] or DTF-IV[1] formats to the necessary probability tables. The major function of this routine is to mix the cross sections stored for each element to form media cross sections and to decompose these cross sections into the individual probability distributions. The Legendre coefficients for each group-to-group transfer may be restored in a permanent storage area after the discrete angles and probabilities have been determined. Output of the cross sections as read (if IRDSG > 0) and as stored (if IPRIN > 0) and the gamma-production cross sections is initiated by this routine. If diagnostic printout of cross-section storage is required a call to XSCHLP (1, 4HJNPT) will give a decimal dump of all cross-section storage and commons.

Called from: XSEC

Subroutines called: READSG, STØRE, LEGEND, ANGLES

Functions used: IABS (library)

Commons required: Blank, LØCSIG, MØMENT, MEANS, RESULT

Variables required: All variables in LØCSIG, see page 4.5-7.

Variables changed: Blank common from ISTART to NMXSEC.

Input read:

MIX, NEL, RHØ — RHØ times the cross section of the element NEL is added to the MIX medium cross section. If NEL is negative, the current mixing operation completes the cross section for that medium. There are NMIX of these cards read.

Significant internal variables:

NDSK is the current number of downscatter groups for starting from present location.

Subroutine LEGEND

Subroutine LEGEND converts Legendre coefficients to moments. The coefficients are given in labelled common MØMENT in the form

$$f_\ell = \int_{-1}^{1} f(\mu)P_\ell(\mu)d\mu \quad \ell = 1,\text{NF} \quad \text{or} \quad f(\mu) = \sum_{\ell=0}^{\text{NF}} \frac{2\ell+1}{2} f_\ell P_\ell(\mu) \quad (f_0 \equiv 1).$$

The output of LEGEND consists of the moments,

$$M_n = \int_{-1}^{1} \mu^n f(\mu)d\mu \quad n = 1,\text{NMØM}.$$

Method: If we let

$$P^{-1}_{n,\ell} = \frac{2\ell+1}{2} \int_{-1}^{1} \mu^n P_\ell(\mu)d\mu.$$

Then by using the fundamental recurrence relation for Legendre polynomials we can derive

$$P^{-1}_{n,\ell} = \frac{1}{2} \int_{-1}^{1} \mu^{n-1}[(2\ell+1)\mu\, P_\ell(\mu)]d\mu$$

$$= \frac{1}{2} \int_{-1}^{1} \mu^{n-1}[(\ell+1)P_{\ell+1}(\mu) + \ell\, P_{\ell-1}(\mu)]d\mu$$

$$= \frac{\ell+1}{2} \int_{-1}^{1} \mu^{n-1}P_{\ell+1}(\mu)d\mu + \frac{\ell}{2} \int_{-1}^{1} \mu^{n-1}P_{\ell-1}(\mu)d\mu$$

$$= \frac{\ell+1}{2\ell+3} P^{-1}_{n-1,\ell+1} + \frac{\ell}{2\ell-1} P^{-1}_{n-1,\ell-1}.$$

Since we have trivially $P^{-1}_{0\ell} = \delta_{0\ell}$ and $P^{-1}_{1\ell} = \delta_{1\ell}$, the coefficients $P^{-1}_{n\ell}$ may easily be computed. Then

$$M_n = \int_{-1}^{1} \mu^n f(\mu)d\mu$$

$$= \sum_{\ell=0}^{n} \frac{2\ell+1}{2} f_\ell \int_{-1}^{1} \mu^n P_\ell(\mu)d\mu$$

$$= \sum_{\ell=0}^{n} P_{n\ell}^{-1} \; f_\ell .$$

Called from: JNPUT

Commons required: MØMENT

Variables required:

NMØM - number of moments given.

F(L), L = 1,NMØM (presumably NF $\geq$ NMØM, no check is made).

Variables changed: XMØMNT(N), N = 1, NMØM.

Significant internal variables:

P1(ℓ) = $P_{n-1,\ell}^{-1}$

P2(ℓ) = $P_{n,\ell}^{-1}$

P10 = $P_{n-1,0}^{-1}$

P20 = $P_{n,\ell}^{-1}$

Limitations: NMØM $\leq$ 24.

Subroutine MAMENT (NMØ)

This routine converts moments to Legendre coefficients. The moments

$$M_n = \int_{-1}^{1} \mu^n f(\mu)d\mu \quad n = 1,NMØ,$$

are given in labelled common MØMENT. The output of the subroutine consists of the same number of Legendre coefficients stored in labelled common MØMENT.

Method: $f_\ell = \int_{-1}^{1} P_\ell(\mu) f(\mu)d\mu$

$$= \sum_{n=0}^{\ell} P_{\ell,n} \int_{-1}^{1} \mu^n f(\mu)d\mu$$

$$= \sum_{n=0}^{\ell} P_{\ell,n} M_n,$$

where the $P_{\ell,n}$ are the coefficients of the ℓth Legendre polynomial,

$$P_\ell(\mu) = \sum_{n=0}^{\ell} P_{\ell,n}\mu^n.$$

Since $P_\ell(\mu) = \dfrac{(2\ell - 1)\mu P_{\ell-1}(\mu) - (\ell - 1)P_{\ell-2}(\mu)}{\ell}$,

$$\sum_{n=0}^{\ell} P_{\ell,n} \mu^n = \left(\frac{2\ell - 1}{\ell}\right)\sum_{n=0}^{\ell-1} P_{\ell-1,n} \mu^{n+1} - \left(\frac{\ell - 1}{\ell}\right)\sum_{n=0}^{\ell-2} P_{\ell-2,n} \mu^n.$$

As this is an identity, we may separately equate the coeff cients of each power of μ giving the relation

$$P_{\ell,n} = \left(\frac{2\ell - 1}{\ell}\right) P_{\ell-1,n-1} - \left(\frac{\ell - 1}{\ell}\right) P_{\ell-2,n}.$$

Since

$P_0(\mu) = 1$ and $P_1(\mu) = \mu$, we have

$P_{0,n} = \delta_{0n}$ and $P_{1,n} = \delta_{1n}$.

Called from: BADMØM

Commons required: MØMENT

Variables required: NMØ, (XMØMNT(N), N=1,NMØ)

Variables changed: (F(L), L=1,NMØ)

Significant internal variables:

P0(n) = $P_{\ell-2,n}$

P1(n) = $P_{\ell-1,n}$

P2(n) = $P_{\ell n}$

P00 = $P_{\ell-2,0}$

P10 = $P_{\ell-1,0}$

P20 = $P_{\ell 0}$

Limitations: NMØ $\leq$ 25.

Subroutine NSIGTA (IGA,JMED,TSIG,PNAB)

This subroutine looks up the total cross section and the non-absorption probability for the group IGA. Modifications have been made to include the use of point, total, and scattering cross sections if IØ6RT is greater than zero. For neutron groups greater than IGQPT and for gamma-ray groups, the multigroup cross sections are used. Point cross sections may be used for neutron-only and neutron-gamma-ray coupled cross sections in either the forward or adjoint solution. Point gamma-ray cross sections are not allowed.

In determining the energy within a multigroup at which the point cross section is determined, an index is chosen randomly between the indices for the corresponding group boundaries. If all supergroup boundaries correspond to some of the multigroup boundaries (with variable supergroup boundaries, this can always be assured), then this is equivalent to assuming a constant flux within the group. Importance sampling of energies within the group is straight-forward, but not provided. See Subroutine GTSCT for description of indexing scheme.

Called from: EUCLID, NXTCØL, User routines

Subroutine called: GTMED

Commons required: Blank, LØCSIG, CTSC1, USER

Variables required:

ISPØRG, ISTART, INABØG, IADJ, NTG from Common LØCSIG

IØ6RT, NNIC, IGQPT, NXSECT, NXPM from Common GTSC1

NQT1 from Common USER

IGA - energy group.

NMED - geometry medium.

Variables changed:

TSIG - total cross section.

PNAB - non-absorption probability.

Subroutine PTHETA (IMED,IGØLD,IGQ,THETA,PMU,NMTG)

This routine calculates the probability per steradian of scattering through an angle whose cosine is THETA for an energy transfer from group IGØLD to other groups. Use is made of the restored Legendre coefficients with the group-to-group transfer incorporated. Thus, evaluation

$$P^{I\to J}(\theta) = \frac{P_s^{I\to J}}{4\pi}\left\{1 + \sum_{\ell=1}^{NM\emptyset M}(2\ell+1) f_\ell^{I\to J} P_\ell(\theta)\right\}$$

where $P_s^{I\to J}$ is the probability of scattering from group I to group J,

$f_\ell^{I\to J}$ is the ℓth Legendre coefficient for scattering from group I to group J,

$P_\ell(\theta)$ is the value of the ℓth Legendre polynomial for an angle whose cosine is θ.

There are NCØEF-1 coefficients restored by JNPUT; i.e., the P_0 table is not restored.

It is assumed that within-group scattering is not zero and is calculated for each entry. An option is provided for calculating the probability of scattering to all other groups or to a set number of downscatter groups.

The following recursion relation is used for calculating the Legendre polynomial:

$$L\,P_L(x) = (2L-1)\,x\,P_{L-1}(x) - (L-1)P_{L-2}(x).$$

Called from: User routines only

Subroutines called: GTMED, XSCHLP

Commons required: Blank, LØCSIG

Variables required:

IMED - geometry medium.

IGØLD - the incoming energy group.

IGQ - the limit of the downscatter for which P(θ) is calculated. That is, P(θ) is determined for group IGØLD to IGQ. If IGQ is zero, full downscatter is assumed and P(θ) is determined for IGØLD to NGP.

THETA - cosine of the scattering angle.

Variables required:

ISTAT, NCØEF, NGP, NTG, NTS, ISPØRG, IDSGØG, INSG, IFSPØG (from common LØCSIG, see page 4.5-7)

Variables changed:

PMU - the probability of scattering through an angle whose cosine is θ; PMU is dimensioned by NMTG.

NMTG - the total number groups to be considered in the problem.

Significant internal variables:

P(K) - Legendre polynomial of order K evaluated at θ.

Limitations: Dimension of 10 for Legendre coefficients. A change in this dimension will allow higher order of expansions.

Function Q(ND,X)

This function subprogram generates $Q_{ND}(X)$ - the value at X of the orthogonal polynomial, Q, of order ND. The recurrence relation for the Q polynomials is employed to generate the function

$$Q_i(x) = (x - \mu_i)\, Q_{i-1}(x) - \sigma^2_{i-1}\, Q_{i-2}(x)$$

$$Q_0(x) = 1$$

$$Q_1(x) = x - \mu_1.$$

Called from: ANGLES, FIND, BADMØM

Commons required: MEANS

Variables required:

ND - the degree of the polynomial desired.

X - the value of the argument desired.

XMU(i) = μ_i, VAR(i) = σ^2_i } in labelled common MEANS.

Variables changed:

Q - the value of the function.

Limitations: ND $\leq$ 14.

Subroutine READSG (LEM,CØF)

The purpose of this routine is to read multigroup cross sections and store them in a buffer region of common. If the flag IDTF is greater than zero, DTF-IV[1] format cross sections may be read; otherwise, the ANISN[2] format in either fixed or free-form is assumed.

The ANISN cross-section format makes use of the repeat feature and the zero feature; thus, there is a mixture of Hollerith and numbers on the card. At present subroutine FFREAD is called to read the data in the IBM-360 version; a slightly modified version of FFREAD could be implemented for other machines rather than their present methods. DTF-IV format does not permit repeats or the zero option, and thus the subroutine reads the card numbers directly into the buffer storage region starting at INPBUF.

If cross sections are read from cards, in the ANISN format, a card sequence check is performed. Columns 73-76 are essentially ignored. Columns 77-80 must either be blank or contain an integer sequence number starting at 1 for each set of cross sections, i.e., for each element and each coefficient. If a card is out of order, the card image is printed and the program continues. This test may be removed by setting IRDSG negative. (This also removes the option of printing the cross section as read.)

If all cross sections which appear on the last card for each coefficient are not needed, a message is printed giving the number of cross sections actually used, and the job is then terminated.

If IXTAPE > 0, cross sections are read from a standard ANISN binary cross section tape. An identification record (4I6,6A8) precedes the cross section for each coefficient. The element identifier must be the fourth integer in the identification record. These identification numbers are required on input card D. Element identifiers do not have to be in the same order as they appear on the cross section tape.

Called from: JNPUT

Subroutines called: FFREAD, XSCHLP

Commons required: Blank, LØCSIC

Variables required: INPBUF, INGP, INDS, KKK, IXTAPE, IDTF (from Common LØCSIG, see page 4.5-7)

LEM - element number for which cross sections are to be read.

CØF - coefficient number for which cross sections are to be read.

INPUT: (INGP*(INDS+3)) values of cross sections for each call.

Significant internal variables:

M - number of cross sections for each coefficient.

NP - number of repeats or zeros for a particular cross section.

Limitations: Card formats must be either ANISN or DTF-IV, or a binary tape may be used. For ANISN-type data on cards, either fixed or free-form may be used.

Subroutine RESTØR (INP, ISP)

This routine is a combination of parts of Subroutines READSG and STØRE. Only the P_0 matrix is read (from tape) and stored either for a forward or adjoint problem. These cross sections are stored temporarily in order to search for maximum number of downscatters.

Called from: GTNDSK

Subroutine called: XSCHLP

Commons required: Blank, LØCSIG

Variables required:

INP - starting location in Blank Common of temporary storage.

ISP - starting location in Blank Common for restored cross sections.

Element ID numbers from Blank Common.

Variables from Common LØCSIG.

Input: P_0 cross sections are read from IXTAPE for each element.

Variables changed: Blank Common.

Subroutine STØRE (IE,IC)

The purpose of subroutine STØRE is to pick up the cross sections for element IE and coefficient IC from the input buffer region and store the total, fission, and downscatter matrix in the temporary storage. Only those parts of the input cross sections that are to be reused are stored. That is, the neutrons may be stripped from a coupled neutron-gamma set, or the gammas may be stripped from a coupled neutron-gamma set. Also, during the restoring the cross sections are transposed if an adjoint solution is desired.

Called from: JNPUT

Commons required: Blank, LØCSIG

Variables required:

IE - element number.

IC - coefficient number.

Cross sections in blank common from INPBUF to INPBUF+INGP*(INDS+3)

INPBUF, NTG, NTS NCØEF, ISPØRT, INFPØG, ISIGØG, INDS, IADJ,

NME (from common LØCSIG, see page 4.5-7)

Variables changed: cross sections in blank common from ISPØRT to ITØTSG

Significant internal variables:

INDX - starting location of downscatter matrix for the IE element and IC coefficient.

IE1 - number of locations to be skipped in the total cross-section array for other elements.

Subroutine XSEC (IADJM,LØCEPR,MEDALB,MEDIA,NLAST,NMGP,NMTG,NLEFT,IØ,IN)

Subroutine XSEC is the primary interface of the cross-section module with the rest of MØRSE.

The function of XSEC is to read the cross-section information defining the number of groups, coefficients, elements, media, etc., and to set up the storage locations required. All variables in Common LØCSIG are defined in Subroutine XSEC. (Three variables are redefined in JNPUT if Legendre coefficients are restored.) After the storage is allocated, Subroutine JNPUT is called and is the executive routine for manipulating the cross sections. If point cross sections are to be used, Subroutine GTSCT is called for the input and storage of these cross sections.

The first medium cross sections are stored from ISTART to ISPØRG + ISTART; each successive medium requires ISPØRG cross sections. The Legendre coefficients are stored after the media cross sections and the point cross sections follow.

Called from: INPUT

Subroutines called: JNPUT, ALBIN, XSCHLP, GTNDSK, XSTAPE, GTSCT

Commons required: Blank, LØCSIG, GTSC1

Variables required:

IADJM - switch indicating that the problem is an adjoint problem if > 0.

LØCEPR - location of energy-biasing parameters; if 0, no energy biasing will be used.

MEDALB - medium number for the albedo scatterer; MEDALB > 0 signals a combined albedo and normal transport problem; = 0 is flag for normal transport only and ALBIN will not be called; < 0 signals an albedo-only problem, normal cross sections will not be read, |MEDALB| is the albedo medium.

MEDIA - number of media for which cross sections are to be read.

NLAST - the last cell used in Blank Common before XSEC was called.

NMGP - the number of primary particle groups for which there are cross sections.

NMTG - total number of groups for which there are cross sections.

NLEFT - dimension of Blank Common. If NLAST > NLEFT, too much storage has been used.

IØ,IN - output and input logical units.

Input: The following four cards are read by Subroutine XSEC*.

First Card† - comment card.

Second Card† - NGP, NDS, NGG, NDSG, INGP, ITBL, ISGG, NMED, NELEM, NMIX, NCØEF, NSCT, ISTAT. For definitions see Common LØCSIG.

Third Card† - IRDSG, ISTR, IFMU, IMØM, IPRIN, IPUN, IDTF, IXTAPE, JXTAPE, IØ6RT, IGQPT. For definitions see Commons LØCSIG and GTSC1.

Fourth Card† - (Omitted if IXTAPE $\leq$ 0) element identifiers of cross sections to be read from tape.

Variables changed:

MEDALB - set to 7777 if there is no albedo surface in problem.

NLAST - the last cell of permanent storage used.

Significant internal variables:

NEC - number of cross sections to be read from tape.

---

*A more detailed description is given in Section 4.3.

†If IXTAPE < 0, cards are required but variables are not used.

Subroutine XSTAPE (ISIG, ITAPE, NLAST, TITLE)

The function of XSTAPE is to either write or read a processed cross-section tape. The processed cross-section tape contains the cross-section title card information, all variables in Common LØCSIG, and the contents of Blank Common from IRSG to NLAST.

If a processed tape is read, the values of the variables normally input are printed after being read from tape for comparison with the input data. However, if an adjoint problem was run in writing the processed tape, there will be differences in the two sets of numbers. The variables in LØCSIG are adjusted to account for a different absolute location in Blank Common.

Called from: XSEC

Commons required: Blank, LØCSIG

Variables required:

ISIG	- switch indicating that a tape is to be written if = 2; a tape is read if ≠ 2.
ITAPE	- logical tape unit of processed cross-section tape.
NLAST	- last location in Blank Common of the cross-section storage.
TITLE	- information from cross-section title card.

4.6. ANALYSIS MODULE

4.6.1. Introduction

SAMBØ* is a package of computer routines which handles most of the drudgery associated with analysis of collisions in a Monte Carlo code. It was written for use with the MØRSE multigroup Monte Carlo code but should be readily adaptable to incorporation into other random walk generating codes. An arbitrary number of detectors, energy-dependent response functions, energy bins, time bins, and angle bins are allowed, with virtually no numerical limitations other than the available core storage. Analysis is divided into (1) uncollided and total response (fluence† integrated over each response function at each detector; (2) fluence versus energy and detector; (3) time-dependent response (time-dependent fluence integrated over each response at each detector); (4) fluence versus time, energy, and detector; and (5) fluence versus angle, energy, and detector. Each quantity in the above arrays is output along with an associated fractional standard deviation.

A simplified flow diagram of a typical Monte Carlo program, interfaced to the SAMBØ package, is shown in Fig. 4.5. Input of parameters and initialization are performed at the beginning of each problem (SCØRIN), each run (STRUN), and each batch (STBTCH). A batch may either be a generation of particles in multiplying systems or a group of histories treated together for calculation of variances. A run is a set of batches. Terminal operations are performed at the end of each batch (NBATCH) and run (NRUN). The above five routines, when called at the appropriate point by the history generation routines, perform most of the necessary bookkeeping functions. Five additional routines are either called by the primary bookkeeping routines or by estimating routines. INSCØR and ENDRUN

---

*Stochastic Analysis Machine for Bookkeeping, if the reader insists on an acronym; the derivation was actually inspired by Samuel Finley Breese Morse's first and third names.

†Current, collision density or other fluence-like quantity may be substituted for fluence at each detector. For example, if current and fluence at a plane are desired, two separate detectors would be used.

Random Walk Generating Package SAMBØ Routines

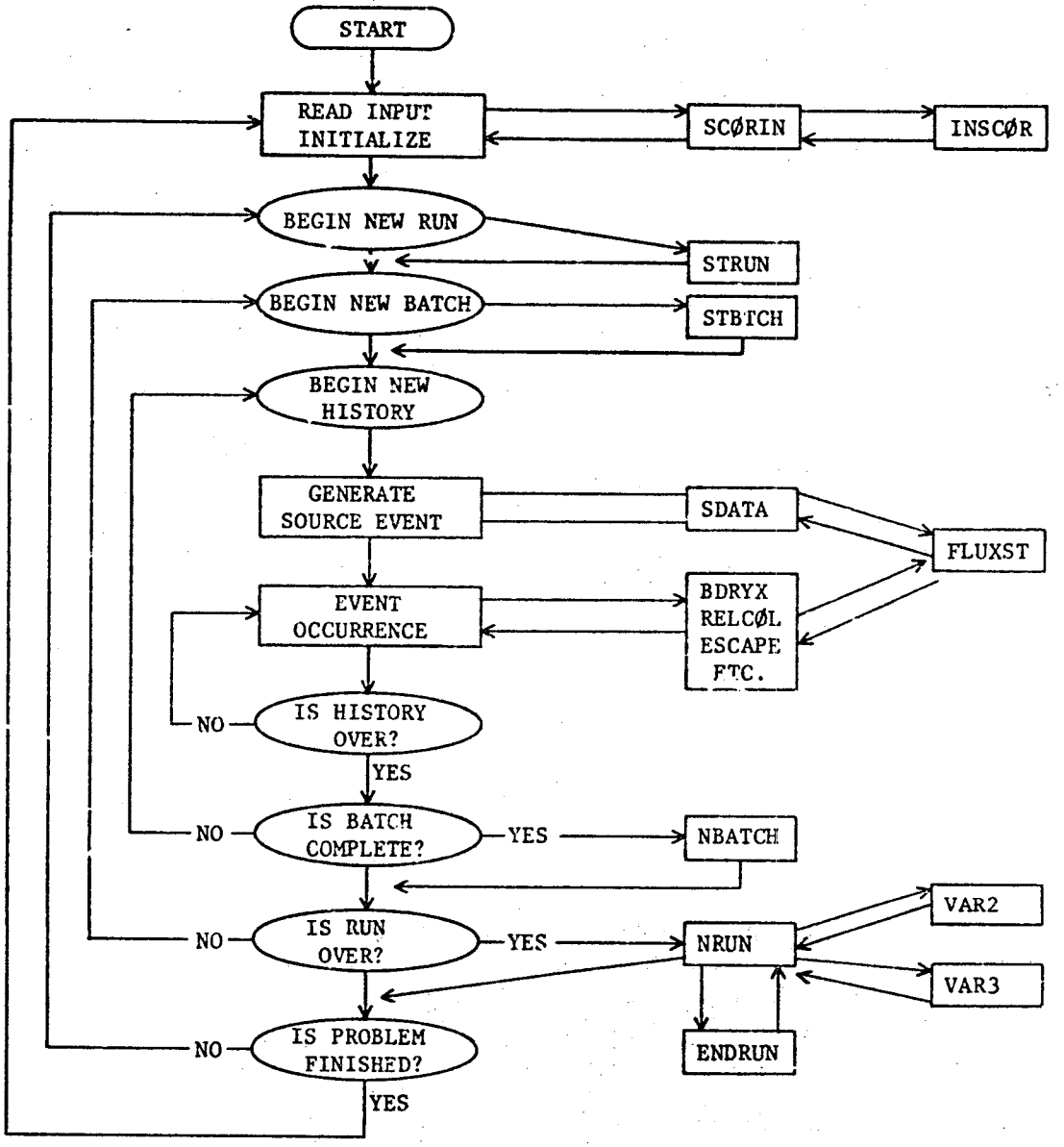


Fig. 4.5. Flow Diagram of Monte Carlo Program Using the SAMBØ Package

are dummies called by SCØRIN and NRUN, respectively, to allow insertion of problem-dependent modifications. VAR2 and VAR3 are called by NRUN to calculate fractional standard deviations in two- and three-dimensional arrays, respectively. FLUXST is the interface between the estimating and bookkeeping routines and is called by the estimating routines to store estimates in the appropriate arrays. The bookkeeping section of the package is described in more detail in Section 4.6.2.

Since so many types of estimators are possible, no attempt is made to provide all-inclusive estimating routines. In lieu of this, samples of typical estimating routines are included, from which the user can construct routines with features close to his needs. Section 4.6.3 describes routines SDATA and BDRYX which implement a surface-crossing estimator of fluence at concentric spheres. Section 4.6.4 includes descriptions of an estimating technique for fluence at a point detector.

4.6.2. Bookkeeping Routines

The interface between the bookkeeping routines in SAMBØ and the random walk routines consists of one labelled common (USER), a specified area of blank common, and calls to four major subroutines. These routines are: (1) SCØRIN, which reads necessary input parameters and sets up room in a specified area of blank common for the necessary arrays; (2) STBTCH, which initializes arrays at the start of each batch; (3) NBATCH, which sums batched estimates and their squares at the end of each batch; and (4) NRUN, which normalizes and outputs results. Three additional routines in this part of the package are VAR2 and VAR3, which are called by NRUN to calculate sample variances based on batch-summed estimates, and STRUN which is called at the beginning of each run (set of batches).

The corresponding interface between the bookkeeping routines in SAMBØ and the estimating routines consists of subroutine FLUXST. This routine is called by the user routines, which calculate estimates from the history parameters to store estimates in the requested arrays in blank common.

The parameters defining the system being treated (i.e., those parameters that do not vary during the walk) by the random walk routines are

placed in common USER before SCØRIN is called. The meanings of these variables are summarized in Table 4.18 and defined more thoroughly below.

AGSTRT, WTSTRT, XSTRT, YSTRT, and ZSTRT represent the starting values of age, statistical weight and position of a discrete source. If the source is not discrete, the values should represent means. The age and position are used to obtain a minimum arrival time (TO) for each detector which defines the lower edge of the first time bin. That is,

TO = AGSTRT + SQRT((XD-XSTRT)**2 + (YD-YSTRT)**2 + (ZD-ZSTRT)**2)/VL

where XD, YD, ZD are the detector position, and VL is the maximum velocity (corresponding to group 1 for a neutron only problem; otherwise the velocity of light).

DFF is a source normalization factor to be used, in adjoint problems, by NRUN to modify all output quantities. It is the factor which normalizes the source distribution function (detector response in the forward problem) so that it is a proper probability density function (p.d.f.). In essence, DFF saves the units inherent in the forward problem detector response. Although its precise definition would depend on the particular code, in most adjoint formulations using point cross sections, DFF would be given by (considering only energy dependence):

$$DFF = \int dE\ P(E),$$

where P(E) is the payoff function in the forward problem and E is the energy variable. This is, of course, the factor needed to alter P(E) to a p.d.f. suitable for use in selecting source particle energies in the adjoint problem. In a multigroup code such as MØRSE, where the scoring function in the forward problem is

$$P_i = \int_{\text{group } i} dE\ P(E) / \int_{\text{group } i} dE,$$

DFF is given by

$$DFF = \sum_i P_i .$$

EBØTN and EBØTG are the lower energy limits of the last neutron† group (group NMGP) and the last gamma-ray group (group NMTG), respectively, for which cross sections, energy bounds, velocities, etc., are available. These seemingly superfluous pieces of information are supplied by MØRSE separately from the upper energy limits of all groups. (The upper energy limits are expected to be in cells 1 to NMTG of blank common, and the corresponding group velocities are expected to be in cells NMTG + 1 to 2*NMTG.) TCUT, IO, I1, and IADJM are defined adequately in Table 4.18.

NGPQT1, -2, and -3 are defined further in Tables 4.13, 4.14, and 4.19 (for a combined neutron and gamma-ray problem). NGPQTN and NGPQTG are the number of neutron and gamma-ray groups, respectively, to be treated in the problem, whereas NMGP and NMTG are the number of primary (neutron) groups and the total number of groups for which cross sections (and some other parameters such as energy limits) are to be provided.

NITS is the initial number of batches requested and NSTRT is the number of independent source particles in each batch.

NLAST and NLEFT define an area of blank common available to the package. Cells NLAST + 1 to NLAST + NLEFT in blank common are assumed to be available.

Subroutine SCØRIN allocates room in this storage area, after receiving values for the variables ND, NNE, NE, NT, NA, NRESP, NEX, NEXND, which are the first eight variables in common PDET described in Table 4.20.

Figure 4.6 is a diagram of this storage area in blank common showing the location of the various arrays. Table 4.21 shows in more detail how these arrays are located. Indices used in the top part of the table are defined in the lower section. For the six sets of three arrays each that contain the estimated quantities, only the first array of each set is described in Table 4.21. This first array, in each case, is used to

---

†"Neutron" and "gamma-ray" are used throughout interchangeably with "primary" and "secondary". This package makes no assumptions about the type of particles being processed except in the definition of VL described previously.

store estimates summed over a batch of histories. The second array of each set uses the mnemonic name prefixed with an S and is used to form the sum over all histories. The third array of each set uses the mnemonic name prefixed with an S and suffixed with a 2 and is used to store sums of squared batch estimates. (See NBATCH, VAR2, and VAR3 writeups for details.)

Following are detailed descriptions of the bookkeeping routines.

Table 4.18. Definitions of Variables in Common USER

AGSTRT	Initial chronological age to be assigned to source particles,
WTSTRT	Initial weight to be assigned to source particles,
XSTRT	Initial x position to be assigned to source particles,
YSTRT	Initial y position to be assigned to source particles,
ZSTRT	Initial z position to be assigned to source particles,
DFF	Normalization for adjoint problems,
EBØTN	Lower energy boundary (eV) of last neutron group (group NMGP),
EBØTG	Lower energy boundary (eV) of last gamma-ray group (group NMTG),
TCUT	Chronological age limit,
IO	Logical unit for output,
I1	Logical unit for input,
IADJM	Adjoint switch (>0 for adjoint problem),
NGPQT1 NGPQT2 NGPQT3	Problem-dependent energy group limits (see text),
NGPQTG	Group number of lowest energy gamma-ray energy group to be treated,
NGPQTN	Group number of lowest energy neutron energy group to be treated,
NITS	Number of batches to be run,
NLAST	Last cell in blank common used by random walk package,
NLEFT	Number of cells in blank common available to user,
NMGP	Number of primary (neutron) energy groups,
NMTG	Total number of energy groups,
NSTRT	Number of source particles for each batch.

Table 4.19. Definitions of Problem-Dependent Energy Group Limits

Forward

NGPQT1 = NGPQTN for neutron only or combined problem,

= NGPQTG for gamma ray only (NGPQTN = 0),

NGPQT2 = NMGP

NGPQT3 = NMGP + NGPQTG for combined problem (NGPQTN*NGPQTG > 0),

= NGPQT1 for neutron only or gamma ray only.

Adjoint

NGPQT1 = NMTG - NMGP,

NGPQT2 = NMTG - NGPQTN,

NGPQT3 = NMTG.

Table 4.20. Definitions of Variables in Common PDET

ND	Number of detectors (≥ 1),
NNE	Number of neutron energy bins,
NE	Total number of energy bins,
NT	Number of time bins,
NA	Number of angle bins,
NRESP	Number of energy response functions (≥ 1),
NEX	Number of extra arrays of length NMTG to be set aside,
NEXND	Number of extra arrays of length ND to be set aside,
NEND	NE*ND,
NDNR	ND*NRESP,
NTNR	NT*NRESP,
NTNE	NT*NE,
NANE	NA*NE,
NTNDNR	NT*ND*NRESP,
NTNEND	NT*NE*ND,
NANEND	NA*NE*ND,
LØCRSP	Location of cell zero of response functions,
LØCXD	Location of cell zero of detector positions,
LØCIB	Location of cell zero of energy bins,
LØCCØ	Location of cell zero of angle (cosine) bins,
LØCT	Location of cell zero of time bins,
LØCUD	Location of cell zero of array UD,
LØCSD	Location of cell zero of array SD,
LØCQE	Location of cell zero of array QE,
LØCQT	Location of cell zero of array QT,
LØCQTE	Location of cell zero of array QTE,
LØCQAE	Location of cell zero of array QAE,
LMAX	Last cell used in blank common,
EFIRST	Upper energy limit of first energy bin,
EGTØP	Upper energy limit of first gamma-ray bin.

Location Labels	Mnemonic Variable Name	Length
NLAST + 1	LABELS	20 * NRESP
LØCRSP		
	RESP	NRESP * NMTG
LØCXD	EXTR	NEX * NMTG
	XD YD ZD RAD TO FACT	6 * ND
LØCIB	EXTR	NEXND * ND
	IB EP DELE	3 * NE omitted if NE=0
LØCCØ	IARR	NMTG
LØCT	CØS	NA
	T DELT	2 * ND * NT
LØCUD		
	UD SUD SUD2	3 * ND * NRESP
LØCSD		
	SD SSD SSD2	3 * ND * NRESP
LØCQE	SUD & SSD Units	20
	QE SQE SQE2	3 * NE * ND
LØCQT	SQE Units	20
	QT SQT SQT2	3 * NT * ND * NRESP
LØCQTE	SQT Units	20
	QTE SQTE SQTE2	3 * NT * NE * ND
LØCQAE	SQTE Units	20
	QAE SQAE SQAE2	3 * NA * NE * ND
LMAX	SQAE Units	20

Fig. 4.6. Layout of SAMBO Analysis Blank Common Storage Area

Table 4.21. Locations of Analysis Arrays in Blank Common

Mnemomic Variable Name	Location in Blank Common BC(I) or NC(I)	Purpose
LABEL (J,NR)	I = NLAST + (NR-1)*20 + J J = 1, 20	80-character Hollerith label for each response function.
RESP (IG,NR)	I = LØCRSP + (NR-1) *NMTG + IG	Value of response function.
EXTR (IG,NX)	I = LØCRSP + NRESP*NMTG + (NX-1)*NMTG + IG	Extra arrays of length NMTG.
XD (ID)	I = LØCXD + ID	x-coordinate of detector ID.
YD (ID)	I = LØCXD + ND + ID	y-coordinate of detector ID.
ZD (ID)	I = LØCXD + 2*ND + ID	z-coordinate of detector ID.
RAD (ID)	I = LØCXD + 3*ND + ID	Distance from source (assumed to be at XSTRT, YSTRT, ZSTRT) to detector.
TO (ID)	I = LØCXD + 4*ND + ID	Minimum flight time to detector.
FACT (ID)	I = LØCXD + 5*ND + ID	Detector-dependent normalization.
EXTR (ID,NXN)	I = LØCXD + 6*ND + (NXN-1)*ND + ID	Extra arrays of length ND.
IB (IE)	I = LØCIB + IE	Energy bin group number.
EP (IE)	I = LØCIB + NE + IE	Lower energy of bin.
DELE (IE)	I = LØCIB + 2*NE + IE	Widths of bin.
IARR (IE)	I = LØCIB + 3*NE + IE	Key to correspondence between analysis group numbers and random walk group numbers.

Table 4.21 (Cont'd.)

Mnemonic Variable Name	Location in Blank Common BC(I) or NC(I)	Purpose
CØS (IA)	I = LØCCØ + IA	Cosine of angle bin limits.
T (IT,ID)	I = LØCT + (ID-1)*NT + IT	Time bin limits.
DELT (IT,ID)	I = LØCT + ND*NT + (ID-1) *NT + IT	Widths of time bins.
UD (NR,ID)	I = LØCUD + (ID-1)*NRESP + NR	Uncollided response.
SD (NR,ID)	I = LØCSD + (ID-1)*NRESP + NR	Total response.
QE (IE,ID)	I = LØCQE + (ID-1)*NE + IE	Energy-dependent fluence.
QT (NR,IT,ID)	I = LØCQT + (ID-1)*NTNR + (IT-1)*NRESP + NR	Time-dependent response.
QTE (IT,IE,ID)	I = LØCQTE + (ID-1)*NTNE + (IE-1)*NT + IT	Time and energy-dependent fluence.
QAE (IA,IE,ID)	I = LØCQAE + (ID-1)*NANE + (IE-1)*NA + IA	Angle- and energy-dependent fluence.

Index	Maximum Value	Purpose
NR	NRESP	Response function.
IG	NMTG	Energy group.
NX	NEX	Extra array (length NMTC).
ID	ND	Detector.
NXN	NEXND	Extra array (length ND).
IE	NE	Energy bin (one or more groups).
IA	NA	Angle bin.
IT	NT	Time bin.

4.6.2. Bookkeeping Routines

Subroutine ENDRUN

This routine is a dummy called by NRUN to provide special problem-dependent output.

Subroutine ENRGYS (E,IB,EP,DELE,IARR)

This routine is called by SCØRIN with the array of upper limits of energy groups, the lower limits of the last primary and secondary energy groups, and the array of desired bins for energy analysis. It then sets up arrays of the energies of the bin boundaries and the size of each bin, and a key to the correspondence between analysis and random walk groups.

Called by: SCØRIN

Routines called: None

Variables required:

- E - NMTG (see page 4.3-1) values of the upper energy limits of each energy group. (input on CARDS F see page 4.3-4)
- IB - NE values of group numbers defining the lower limit of each energy bin (input on CARDS HH, see page 4.3-15).
- NE - total number of energy bins.
- NNE - number of energy bins for primary particles.
- NMGP, NMTG, IADJM, EBØTG, EBØTN - for definitions, see common USER, page 4.6-7 .

Variables modified:

- EP - NE values of the energies corresponding to lower edges of energy bins.
- DELE - NE values of the widths of the energy bins.
- EFIRST - upper energy of highest energy primary bin (or secondary, if no primaries).
- EGTØP - upper energy of highest energy secondary bin, if used.
- IARR - NMTG values of a key to correspondence between analysis group numbers and random walk group numbers.

Commons required: USER, PDET.

Subroutine FLUXST (I,IG,FLUX,AGE,CØS,SWITCH)

This routine is given FLUX, a fluence or current-like estimate to be stored. The parameters I, IG, AGE, and CØS represent the position, energy, age, and cosine of an angle characterizing the estimate. SWITCH is an integer specifying which arrays are to be stored in. Using IG, AGE, and CØS the routine determines the appropriate energy, time, and angle bins (indices J, K, and L). These analyses are bypassed if NE, NT, or NA, respectively, is zero. If IG or CØS are out of the allowed range, an error call results. If AGE is beyond the time bin limits, the last time bin for that detector is extended to accommodate the score and execution continues.

In the fluence-like arrays (QE, QTE, and QAE) FLUX is stored directly. In the response arrays (UD, SD, and QT) FLUX*RESP(IG) is stored, where RESP(IG) represents each of the response functions for energy group IG, as input by the user.

Called from: User-estimating routine

Routines called:

HELP (user-provided dump subroutine; called if IG or CØS is out of the possible range).

ERRØR (library subroutine).

Commons: Blank, USER, PDET.

Variables required:

I	- detector index.
IG	- energy group.
FLUX	- fluence-like estimate to be stored.
AGE	- chronological age.
CØS	- cosine of angle of interest.
SWITCH	- integer switch. > 0, store in all relevant arrays; = 0, store in all relevant arrays, except UD; < 0, store in array UD only.
NMTG	- (for definition, see common USER, page 4.6-7).

NA
ND
NE
NT
LØCT
NANE
NTNE
NTNR
LØCCØ
LØCIB
LØCQE
LØCQT
LØCSD
LØCUD
NRESP
LØCQAE
LØCQTE
LØCRSP

for definitions, see common PDET, page 4.6-9 .

Significant internal variables:

SCØRE (=FLUX*RESP(IG))

Variables modified:

Blank common arrays UD (if SWITCH ≠ 0)
SD
QE (if NE > 0)
QT (if NT > 0)
QTE (if NE > 0 and NT > 0)
QAE (if NE > 0 and NA > 0)

Subroutine INSCØR

This routine is a dummy called by SCØRIN to provide special problem-dependent analysis input.

Subroutine NBATCH (NSØRC)

This routine is called at the end of each batch to perform the sums needed for the estimated means and for calculation of batch statistics. (If the system is multiplying, batches become interpreted as generations, and statistics are then based on multiple runs.) Provision is made for batches of different sizes through the argument NSØRC. Because of this, the summation of the square of the accumulated estimate is divided by NSØRC, the number of particles starting the batch. (See VAR2 and VAR3 writeups for statistical formulae.)

Called by: User at end of batch (or run).

Routines called: None

Variables required:

NSØRC - number of particles beginning the batch (or other statistically-independent grouping).

NA, ND, NE, NT, NANE, NDNR, NEND, NTNE, NTNR, LØCQE, LØCQT, LØCSD, LØCUD, NRESP, LØCQAE, LØCQTE, NANEND, NTNDNR, NTNEND — for definitions, see common PDET, page 4.6-9 .

UD, SD, QE, QT, QTE, QAE — Arrays in blank common, see Fig. 4.6 and Table 4.21.

Variables modified:

SUD, SUD2 SSD, SSD2 SQE, SQE2 SQT, SQT2 SQTE, SQTE2 SQAE, SQAE2	arrays of sums over all batches and sums of squared batch estimates, see Fig. 4.6 and related text.

Commons required: Blank, PDET

Subroutine NRUN (NBATS, NRUNS)

This routine is called at the end of each run (or the end of a problem). The calculated quantities are normalized and output, along with fractional standard deviations (f.s.d.). Much of the coding is complicated because there are no restrictions on the sizes of the scoring arrays packed in blank common. NRUN must therefore decide which part of an array to put on each page, etc.

Called by: User upon problem completion.

Routines called:

DATE (A,I) - a subroutine (see Section 4.4, page 23 for detailed description) which returns a hollerith string in array A (I 4-byte words long) containing the day of the week and the date.

ENDRUN - a dummy subroutine called as the last step in NRUN which may be used to provide problem-dependent output not provided by NRUN.

HELPER† - a subroutine (see Section 4.8, page 13 for detailed description) used to output a part of an array in decimal form. It is called by NRUN to output, if used, the extra arrays starting at LOCSD + 6*ND + 1 in blank common.

INSERT† - library subroutine at Oak Ridge National Laboratory; used to insert a hollerith string in another string.

INTBCD† - library subroutine at ORNL; converts a 4-byte integer to a hollerith string.

VAR2, VAR3 - calculate f.s.d.'s; see writeup, page 4.6-27-29)

Variables required:

NBATS - number of batches completed. (Note that NITS in common USER is the requested number of batches, not necessarily the actual number completed.)

NRUNS - number of runs remaining plus one, or negative of the number of runs completed when an execution time kill occurs.

---

†Not used by CDC-6600 or UNIVAC-1108 versions of this subroutine.

IO
DFF
IADJM
NLAST
NSTRT

- for definitions, see common USER, page 4.6.7 .

Essentially all variables in common PDET, page 4.6-9 .

Variables modified:

SUD
SSD
SQE
SQT
SQTE
SQAE

- these arrays in blank common are normalized and printed.

SUD2
SSD2
SQE2
SQT2
SQTE2
SQAE2

- these arrays in blank common are converted to f.s.d.'s and printed.

Commons required: Blank, USER, PDET.

Subroutine SCØRIN

This routine is called by INPUT2 to input data necessary to define the scope of the analysis. Some initialization and problem-dependent preparation is also performed. As its last step, SCØRIN calls INSCØR, a dummy routine for input of additional problem-dependent data.

Called by: INPUT2

Routines called:

INSCØR - see above.

ENRGYS - given arrays of upper limits of energies of groups (eV) and the desired bins for analysis, sets up arrays of limits of bins (eV) and the size of all bins.

Commons: USER, PDET.

The following data are read from cards by SCØRIN:

CARD AA (20A4)

Title information - will be immediately output.

CARD BB (14I5)

ND - number of detectors (must be $\geq$ 1).

NNE - number of primary particle (neutron) energy bins to be used.

NE - total number of energy bins.

NT - number of time bins for each detector, (may be negative, in which case |NT| values are to be read and used for every detector).

NA - number of angle bins.

NRESP - number of energy-dependent response functions to be used (must be $\geq$ 1).

NEX - number of extra arrays of size NMTG (see common USER writeup, page 4.6-7) to be set aside (useful, for example, as a place to store an array of group-to-group transfer probabilities - see RELCØL writeup for example).

NEXND - number of extra arrays of size ND to be set aside (useful, for example, as a place to store detector-dependent counters - see BDRYX writeup for example).

Note that if ND or NRESP $\leq$ 0 on input, they will be set to one. If NA or NE $\leq$ 1, they will be set to zero. If NT = $\pm$ 1, it will be set to zero.

CARDS CC (3E10.4) (ND cards will be read)

X,Y,Z - detector location. (If other than point detectors are desired, the point locations must still be input and can be combined with additional data built in to user routines to fully define each detector.)

Note that the distance between the above points and the XSTRT. YSTRT, ZSTRT values and the initial age, AGSTRT (see common USER writeup, page 4.6-7) will be used to define the lower limit of the first time bin.

CARD DD (20A4)

Units for SUD and SSD array printouts. Will be used in columns 54 through 133 of the title for the print of these arrays.

CARD EE (20A4)

Identification for each response function. Will be used as title in printout.

CARDS FF (7E10.4)

Response function values. NMTG values will be read in each set of FF cards. Input order is from energy group 1 to NMTG (order of decreasing energy).

Note: Cards EE and FF are read in the following order: EE,FF1,...FFN, EE,FF1,...FFN, etc.

CARD GG (20A4) (omit if NE $\leq$ 1)

Units for title of SQE array printout.

CARDS HH (14I5) (omit if NE $\leq$ 1)

Energy group numbers defining lower limit of energy bins (in order of increasing group number). The NNE (if > 0) entry must equal NGPQTN; the NR entry must be set to NMGP + NGPQTG for a combined problem, or else NGPQTG or NGPQTN.

CARD II (20A4) (omit if NT $\leq$ 1)

Units for title of SQT array printout.

CARD JJ (20A4) (omit if NT $\leq$ 1 or NE $\leq$ 1)

Units for title of SQTE array printout.

CARDS KK (7E10.4) (omit if $|NT| \leq 1$)

NT values of upper limits of time bins for each detector (in order of increasing time and detector number). The values for each detector must start on a new card. $|NT|$ values only are read if NT is negative. They are then used for every detector.

CARD LL (20A4) (omit if NA $\leq$ 1)

Units for title of SQAE array printout.

CARD MM (7E10.4) (omit if NA $\leq$ 1)

NA values of upper limits of angle bins (actually cosine bins; the NAth value must equal one).

Subroutine STBTCH (NBAT)

The arrays used to accumulate estimated quantities during a batch are zeroed by this routine. In addition, if NBAT = 0 indicating the first batch in a run is about to begin, all arrays are zeroed which accumulate estimates and squared estimates over batches.

Called by: USER in BANKR usually.

Routines called: ERRØR (library)

Variables required:

NBAT - batch number less one.

Most of the variables in common PDET, see page 4.6-9 .

Variables modified:

Arrays in blank common, see Fig. 4.6 and Table 4.21.

Commons required: Blank, PDET.

Subroutine STRUN

This routine is called at the beginning of each set of NITS batches and is normally used only for problems like time-dependent fissioning systems.

Subroutine VAR2 (SX, SX2, M1, M2, NBAT, NPART)

This routine calculates variances and f.s.d.'s for batch statistics allowing for unequally weighted batches. The formula for variance of the mean is:

$$\sigma_{\bar{x}}^2 = \frac{1}{(N-1)} \left[\frac{1}{n} \sum_{i=1}^{N} n_i x_i^2 - \frac{1}{n^2} \left[\sum_{i=1}^{N} n_i x_i \right]^2 \right],$$

where N = number of batches,

n = total number of independent histories,

n_i = number of independent histories in i^{th} batch,

x_i = accumulated estimate in i^{th} batch.

Note that:

$$n = \sum_{i=1}^{N} n_i$$

$$x_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x_{ij}$$

where x_{ij} is the estimate from the j^{th} history in the i^{th} batch,

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{N} n_i x_i$$

where $\bar{x}$ is the mean, averaged over n histories.

The fractional standard deviation is

$$\text{f.s.d.} = \sqrt{\sigma_{\bar{x}}^2} \Big/ \bar{x} .$$

Note that the routine must be called before the array SX ($=n\bar{x}$) is normalized.

Called from: NRUN

Routines called: SQRT (library function),
ABS (library function).

Variables required:

SX (I,J) - array of values of $\overline{nx} = \sum_{i=1}^{N} n_i x_i$.

SX2 (I, J) - array of values of $\sum_{i=1}^{N} n_i x_i^2$.

M1 - maximum value of I subscript.

M2 - maximum value of J subscript.

NBAT - N.

NPART - n.

Variables modified:

SX2 (I, J) changed to f.s.d.

Subroutine VAR3 (SX, SX2, M1, M2, M3, NBAT, NPART)

Identical to VAR2 except that SX and SX2 are three-dimensional with dimensions M1, M2, and M3, respectively.

Table 4.22. BANKR Arguments

BANKR Argument	Called From	Location of call in walk
-1	MØRSE	After call to INPUT - to set parameters for new problem.
-2	MØRSE	At the beginning of each batch of NSTRT particles.
-3	MØRSE	At the end of each batch of NSTRT particles.
-4	MØRSE	At the end of each set of NITS batches - a new problem is about to begin.
1	MSØUR	At a source event.
2	TESTW	After a splitting has occurred.
3	FPRØB	After a fission has occurred.
4	GSTØRE	After a secondary particle has been generated.
5	MØRSE	After a real collision has occurred - post-collision parameters are available.
6	MØRSE	After an albedo collision has occurred - post-collision parameters are available.
7	NXTCØL	After a boundary crossing occurs (the track has encountered a new geometry medium other than the albedo or void media).
8	NXTCØL	After an escape occurs (the geometry has encountered medium zero).
9	MØRSE	After the post-collision energy group exceeds the maximum desired.
10	MØRSE	After the maximum chronological age has been exceeded.
11	TESTW	After a Russian roulette kill occurs.
12	TESTW	After a Russian roulette survival occurs.
13	GSTØRE	After a secondary particle has been generated but no room in the bank is available.

4.6.3. Interface Routines

Subroutine BANKR (NBNKID)

The function of BANKR is to call analysis a d diagnostic subroutines as specified by the user. The particular subroutines called in the analysis module are determined by the index NBNKID. In most problems: BANKR (-4) calls NRUN: BANKR (-3) calls NBATCH: BANKR (-2) calls STBTCH: BANKR (-1) calls STRUN and HELP. Also, if the user is using a boundary crossing estimator, BANKR (1) calls SDATA: and BANKR (7) calls BDRYX while any other values of NBNKID result in a return. BANKR may be called with as many as 17 values of the argument index to direct the analysis. It should be noted that not all the BANKR calls listed in Table 4.22 are actually programmed in the code. The user may have to add these calls for his special purposes. The user also must revise BANKR to call the proper routines to analyze the collision types 1 through 13 that he is interested in.

A version of BANKR that writes a collision tape similar to that written by Ø5R is also available. There are 36 possible variables that may be written on the tape for each of the 13 types of events. The use of the tape-writing version of BANKR is not encouraged but it is provided for that occasional circumstance where it is advantageous.

Called from: MSØUR, FPRØB, MØRSE, NXTCØL, TESTW, GSTØRE

Subroutines called: STRUN, STBTCH, NBATCH, NRUN, HELP, SDATA, BDRYX, RELCØL, etc.

Commons required:

- NBNKID - an index which identifies the type of collision and/or subroutine called (NBNKID = -4, -3, -2, -1, 1, 2, ... 13; see explanation of common APØLLØ - Table 4.4 to learn what collision types 1-13 are).
- NITS - number of batches to be run.
- ITERS - number of batches which remain to be processed.
- NQUIT - number of runs remaining plus one (set to negative of the number of runs completed, when an execution time kill occurs).

NMEM - number of particles which remain to be processed in a given batch.

Significant internal variables:

NBAT - the batch number less one.

NSAVE - the number of particles starting the current batch.

4.6.4. Surface-Crossing Estimators

In many problems, the desired basic quantity is a particle fluence averaged over a surface. A common procedure is to score the weight of the particle divided by the absolute value of the cosine of the angle between the particle track and the surface normal. The estimate to the average fluence is then the sum of the scores divided by the area of the surface considered.

It is well known that the grazing angles in this procedure result in an infinite variance. Clark[15] showed that these grazing angles produce higher order effects in variance than in fluence, thereby justifying an excluded band of angles near grazing along with a nonstochastic estimate of the excluded component. A brief justification of a reasonable procedure, based on Clark's derivation, follows.

Defining, at a particular point on the surface, n, the normal to the surface,

$\phi(w)$, the fluence per steradian in direction w, and

$\mu = w \cdot n$, the cosine of the angle between w and n,

the fluence at this point is

$$\phi = \pi \int_0^1 \phi(\mu)\, d\mu = \pi \int_0^1 \mu\phi(\mu) \frac{1}{\mu}\, d\mu.$$

(If these quantities vary over the surface, they must be averaged over the surface to obtain the desired quantity.)

Since the particle weights crossing the surface in the Monte Carlo are samplings from the angular current, $\mu\phi(\mu)$, it is clear from the above equation that $1/\mu$ is a valid estimator for the fluence.

The variance of the $1/\mu$ estimate of fluence is

$$\sigma^2_{1/\mu} = \left\langle \frac{1}{\mu^2} \right\rangle - \left\langle \frac{1}{\mu} \right\rangle^2$$

$$= \frac{\int_0^1 \mu\phi(\mu) \frac{1}{\mu^2}\, \phi\mu}{\int_0^1 \mu\phi(\mu)\, \phi\mu} - \left[\frac{\int_0^1 \mu\phi(\mu) \frac{1}{\mu}\, \phi\mu}{\int_0^1 \mu\phi(\mu)\, \phi\mu} \right]^2$$

$$= \frac{wj - \phi^2}{j^2}$$

where

$$w = 2\pi \int_0^1 \frac{1}{\mu} \phi(\mu) d\mu, \text{ and}$$

$$j = 2\pi \int_0^1 \mu\phi(\mu) d\mu \quad \text{(the total current).}$$

Considering a simple linear angular flux,

$$\phi(\mu) = g_0 + g_1\mu,$$

and excluding the range $0 < \mu < \varepsilon$ from consideration, one may obtain for the expected value of the fluence estimate

$$\phi(\varepsilon) = 2\pi \left[g_0(1-\varepsilon) + g_1 \frac{(1-\varepsilon^2)}{2} \right]$$

and for the dominant term in the variance,

$$w(\varepsilon) = 2\pi \left[g_0 \ell n \frac{1}{\varepsilon} + g_1(1-\varepsilon) \right] .$$

If, in order to estimate the excluded contribution, one assumes a constant angular flux in the excluded range ($g_0 = 1/2\pi$ and $g_1 = 0$), then the mean fluence excluded is

$$2\pi \int_0^{\varepsilon} \frac{1}{2\pi} d\mu = \varepsilon.$$

However, if the constant angular fluence assumption is wrong, and in actuality has no constant term ($g_0 = 0$ and $g_1 = 1/\pi$), the mean fluence excluded is

$$2\pi \int_0^{\varepsilon} \frac{\mu}{\pi} d\mu = \varepsilon^2.$$

Thus, in this sample the estimated fluence will tend to $1-\varepsilon^2+\varepsilon$ rather than 1. In most practical cases an error of this magnitude is quite acceptable and insures a finite variance.

As an alternate procedure to simply adding ε to the estimated answer to correct for the excluded band, one may score the expected value of $1/\mu$ for every occurrence of $\mu<\varepsilon$. If, again, a constant angular fluence in $0 < \mu < \varepsilon$ is assumed, the p.d.f. of arrivals is $2\mu/\varepsilon^2$. The expected value of $1/\mu$ is then

$$\frac{2}{\epsilon^2} \int_0^{\epsilon} \mu \frac{1}{\mu} \, d\mu = \frac{2}{\epsilon}.$$

This is the technique actually used in the surface-crossing routine, BDRYX, described below. BDRYX is intended to be called from the random walk module at every entry of a particle track into a different geometry medium. SDATA is to be called for every source event. Both of these routines interface to the bookkeeping package through the routine FLUXST, which stores each estimate in the proper locations.

Subroutine BDRYX

This routine is called whenever the geometry tracking routines encounter a change in media. If the source-to-collision distance corresponds to a detector position, the reciprocal of the cosine of the angle from the radius vector to the particle direction is used as a fluence estimate. FLUXST is called to store the estimate in the appropriate arrays.

Called by: User at medium boundary intersections (in Subroutine BANKR when NBNKID = 7) or in some instances when a particle escapes (in Subroutine BANKR when NBNKID = 8).†

Routines called:

- ERRØR - (library).
- ABS - (library function).
- HELP - a dump subroutine (see Section 4.8, page 11 for detailed description).
- FLUXST

Variables required:

- X,Y,Z - boundary crossing position.
- ND - number of detectors.
- LØCXD - location of cell zero in blank common of the detector positions.
- U,V,W - particle direction vector.
- WATE - current particle weight.

Significant internal variables:

- R21 - radial distance to boundary crossing.
- R2 - 99% of R21.
- R22 - 101% of R21.
- CØS - cosine of angle between particle direction and radius vector.
- ABC - ABS(CØS).
- CØN - fluence estimate.

Commons: Blank, USER, NUTRØN, PDET.

---

†In the escape case, WTBC must be used in place of WATE.

Subroutine SDATA

Called from the walk module for each source collision, this routine calculates uncollided fluence for each detector. The version described here assumes an infinite medium, but generalization is simple (identical to point detector SDATA). FLUXST stores these estimates in the arrays provided for uncollided quantities.

Routines called:

EXP - (library function).

NSIGTA - (provides TSIG, given IG and NMED - see below).

FLUXST

Variables required:

IG - energy group index.

NMED - medium number.

TSIG - macroscopic total cross section provided by NSIGTA.

ND - number of detectors.

LØCXD - location of cell zero in blank common of detector positions.

WATE - neutron weight.

Commons: Blank, PDET, USER, NUTRØN.

4.6.5. Point Detector Estimators

It is frequently desirable to use Monte Carlo techniques to estimate particle fluence, or a fluence-like quantity, at a point in space. The basic method described here is the "next-event" estimator which scores, from each collision point, the probability of the next event being at the detector. The theoretical basis of the technique is reviewed briefly.

The random walk process in a Monte Carlo code generates a sequence of collisions which are samples from the event density p.d.f. E(x) defined by

$$E(x) = S(x) + \int E(y)K(y,x)\,dy, \tag{4.1}$$

where S(x) is the p.d.f. of particle births, K(y,x) is the conditional p.d.f. of events at x, given that an event occurred at y, and x and y are points in phase space.

If S(x) and K(y,x) are given in terms of the coordinates of particles entering an event (collision or birth), then the flux at a point ($\underline{r}$) can be given by

$$\phi(\underline{r}) = \int dv' \int d\underline{r}' \int dv \int d\underline{\Omega} E(r',v,\underline{\Omega}) \left[p\left[\frac{\underline{r}-\underline{r}'}{|\underline{r}-\underline{r}'|} \cdot \underline{\Omega}; v \right] \frac{e^{-\Sigma(v')|\underline{r}-\underline{r}'|}}{|r-r'|^2} \right] \tag{4.2}$$

where $E(\underline{r}',v,\underline{\Omega})d\underline{r}'dvd\underline{\Omega}$ is the density of particles entering collision in volume $d\underline{r}'$ with speed in dv and direction in $d\underline{\Omega}$, $p(\mu,v)$ is the probability per steradian of scattering a particle of speed v through an angle $\cos^{-1}\mu$, v' is the outgoing speed, and $\Sigma(v')$ is the total macroscopic cross section for speed v'. If more than one medium is present, $\Sigma(v')|\underline{r}-\underline{r}'|$ is replaced by the total number of mean free paths between $\underline{r}'$ and $\underline{r}$.

Since the random walk generates selections from $E(\underline{x})$, estimates to $\phi(\underline{r})$ may be obtained by evaluating the quantity in brackets at all collision sites $\underline{x}$.

Descriptions of subroutines SDATA and RELCØL, using the next-event estimator to score from source and collision events, respectively, follow.

Subroutine RELCØL

The fluence estimate, from the normal collisions in RELCØL, is given by

$$\frac{WATE \cdot e^{ARG} \cdot p(C\emptyset SSCT, IG\emptyset)}{R^2} \qquad (4.3)$$

where WATE is the statistical weight of the particle leaving the collision, ARG is the negative of the number of mean free paths from collision to detector, R is the distance from collision to detector, p(CØSSCT,IGØ) is the probability, per steradian, for a particle with energy in the incoming group IGØ scattering to a lower energy group through angle $\cos^{-1}$(CØSSCT) and CØSSCT is the cosine of the angle between the incoming direction and the line from the collision site to the detector. Note that an array is generated by PTHETA which gives the probability, per steradian, of scattering from group IGØ to groups IGØ to IGQUIT through an angle $\cos^{-1}$(THETA).

Called by: User at real collisions (in Subroutine BANKR when NBNKID = 5).

Routines called:

SQRT	- (library function).
PTHETA	- calculates array p, defined above.
EUCLID	- determines ARG.
NSIGTA	- determines total cross section.
FLUXST	

Variables required:

X,Y,Z	- collision site.
IGØ	- incoming energy group index.
LØCXD	- location in blank common of cell zero of detector position.
ND	- number of detectors.
NMED	- medium of X,Y,Z.
NMTG	- number of energy groups.
WATE	- statistical weight leaving X,Y,Z.
AGE	- chronological age at X,Y,Z.
NEX	- must be $\geq$ 1 since an array of this size is required for the probabilities calculated by PTHETA.

Commons required: Blank, USER, PDET, NUTRØN.

Subroutine SDATA

Estimates of fluence at several point detectors are made for each source event. The source is assumed to emit isotropically and is located at the point X,Y,Z. The contribution is

$$\frac{(\text{WATE})e^{\text{ARG}}}{4\pi R^2}$$

where WATE is the statistical weight of the source particle,

ARG is the negative of the number of mean free paths from source to detector,

R is the distance from source to detector.

Called by: User at source events (in Subroutine BANKR when NBNKID = 1).

Routines called:

EUCLID - Routine which determines ARG.

FLUXST

EXP - (library function).

SQRT

Variables required:

ND - number of detectors.

LØCXD - location of cell zero in blank common of detector positions.

IG - energy group index.

Commons: Blank, USER, PDET, NUTRØN.

4.7. CG - THE COMBINATORIAL GEOMETRY MODULE

4.7.1. Body Types

Combinatorial geometry (CG) describes general three dimensional material configurations by considering unions, differences, and intersections of simple bodies such as spheres, boxes, cylinders, etc. In effect, the geometric description subdivides the problem space into unique zones.* Each zone is the result of combining one or more of the following geometric bodies.

1. Rectangular Parallelepiped (RPP)
2. Box (An RPP randomly oriented in space)
3. Sphere
4. Right Circular Cylinder
5. Right Elliptical Cylinder
6. Truncated Right Angle Cone
7. Ellipsoid
8. Right Angle Wedge
9. Arbitrary Convex Polyhedron of 4, 5, or 6 sides

Body types 2-9 may be arbitrarily oriented with respect to the x, y, z coordinate axes used to determine the space. Body 1, a special body described below, must have sides which are parallel to the coordinate axes.

4.7.2. Introduction

The basic technique for the description of the geometry consists of defining the location and shape of the various zones in terms of the intersections and unions of the geometric bodies. A special operator notation involving the symbols (+), (-), and (ØR) is used to describe the intersections and unions. These symbols are used by the program to construct information relating material descriptions to the body definitions.

---

*To avoid confusion between importance regions and combinatorial geometry regions, we depart from previous combinatorial geometry descriptions and use the term zone to indicate a combinatorial geometry region which is designated by the variable IR. The term region is reserved for an importance region. Thus the zone index is IR.

If a body appears in a zone description with a (+) operator, it means that the zone being described is wholly contained in the body. If a body appears in a zone description with a (-) operator, it means that the zone being described is wholly outside the body. If the body appears with an (ØR) operator, it means that the zone being described includes all points in the body. ØR may be considered as a union operator. In some instances, a zone may be described in terms of subzones lumped together by (ØR) statements. Subzones are formed as intersects and then the zone is formed by a union of these subzones. When (ØR) operators are used there are always two or more of them, and they refer to all body numbers following them, either (+) or (-). That is, all body numbers between "ØR's" or until the end of the zone cards for that zone are intersected together before ØR's are performed.

Techniques for describing a particular geometry are best illustrated by examples. Consider an object composed of a sphere and a cylinder as shown in Fig. 4.7. To describe the object, one takes a spherical body (2) penetrated by a cylindrical body (3) (see Fig. 4.7). If the materials in the sphere and cylinder are the same, then they can be considered as one zone, say zone I (Fig. 4.7c). The description of zone I would be

$$I = ØR + 2ØR + 3.$$

This means that a point is in zone I if it is either inside body 2 or inside body 3.

If different materials are used in the sphere and cylinder, then the sphere with a cylindrical hole in it would be given a different zone number (say J) from that of the cylinder (K).

The description of zone J would be (Fig. 4.7d):

$$J = +2 - 3.$$

This means that points in zone J are all those points inside body 2 which are not inside body 3.

The description of zone K is simply (Fig. 4.7e):

$$K = +3.$$

That is, all points in zone K lie inside body 3.

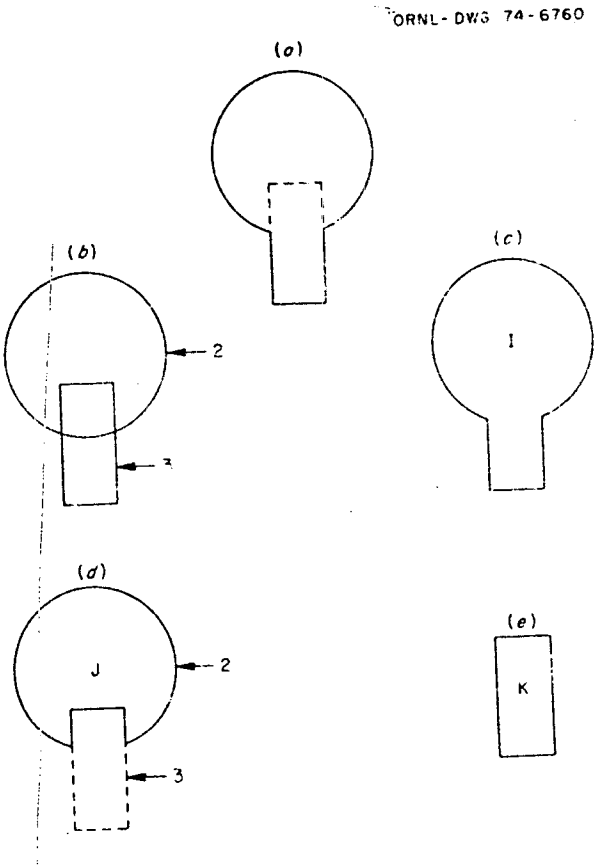


Fig. 4.7. Examples of Combinatorial Geometry Method.

Combinations of more than two bodies and similar zone descriptions could contain a long string of (+), (-), and (ØR) operators. It is important however to remember that every spatial point in the geometry must be located in one and only one zone.

As a more complicated example of the use of the (ØR) operator, consider the system shown in Fig. 8 consisting of the shaded zone A and the unshaded zone B. These zones can be described by the two BØX's, bodies 1 and 3, and the RCC, body 2. The zone description would be

A = +1 +2

and

B = ØR + 3 - 1ØR + 3 - 2.

Notice that the ØR operator refers to all following body numbers until the next ØR operator is reached.

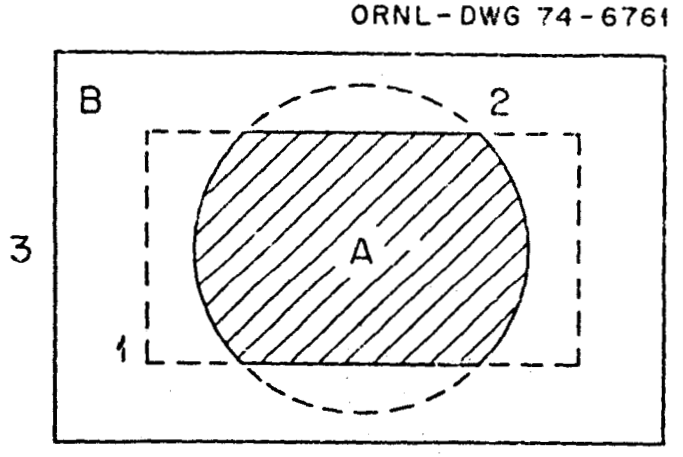


Fig. 4.8. Use of ØR Operators.

The geometry must be specified by establishing two tables. The first table describes the type and location of the set of bodies used in the geometrical description. The second table identifies the physical zones in terms of these bodies. The input routine processes these tables to put the data in the form required for ray tracing. Because the ray tracing routines cannot track across the outermost body, all of the zones must be within a surrounding external void so that all escaping particles are absorbed. Also no point may be in more than one zone. This geometry package is in double precision on the IBM-360 system but remains in single precision on UNIVAC-1108 and CDC-6600 versions. Because of the change in precision, variables in commons PAREM and ØRGI had to be reordered in the IBM-360 version.

4.7.3. Description of Body Types

The information required to specify each type of body is as follows:

a. Rectangular Parallelepiped (RPP)

Specify the minimum and maximum values of the x, y, and z coordinates which bound the parallelepiped.

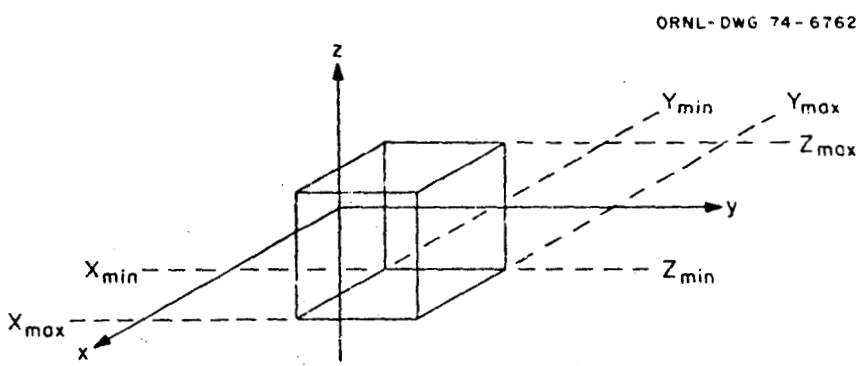


Fig. 4.9. Rectangular Parallelepiped (RPP).

b. Sphere (SPH)

Specify the vertex $\underline{V}$ at the center and the scalar, R, denoting the radius.

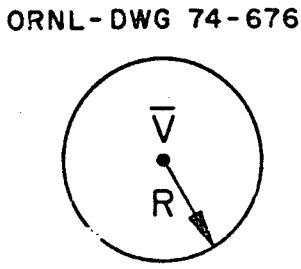


Fig. 4.10. Sphere (SPH).

c. Right Circular Cylinder (RCC)

Specify the vertex $\underline{V}$ at the center of one base, a height vector, $\underline{H}$, expressed in terms of its x, y, and z components, and a scalar, R, denoting the radius.

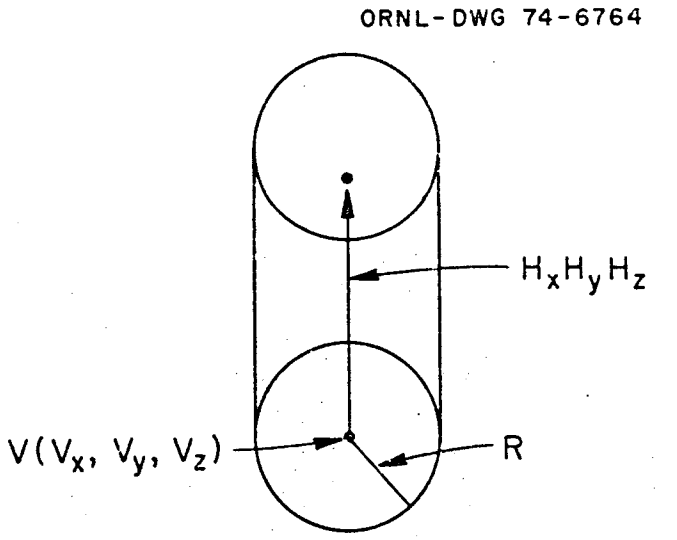


Fig. 4.11. Right Circular Cylinder (RCC).

d. Right Elliptical Cylinder (REC)

Specify coordinates of the center of the base ellipse, a height vector, and two vectors in the plane of the base defining the major and minor axes.

ORNL-DWG 74-6765

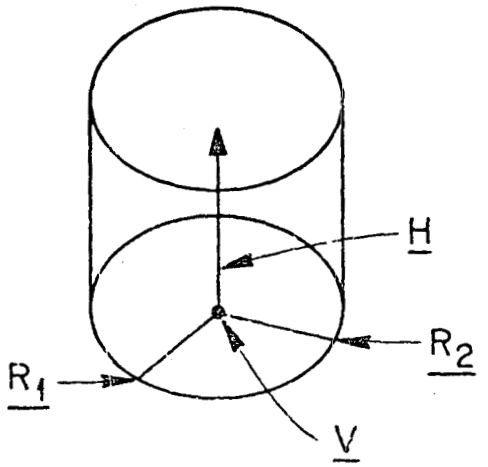


Fig. 4.12. Right Elliptical Cylinder (REC).

e. Truncated Right Angle Cone (TRC)

Specify a vertex V at the center of the lower base, the height vector, H, expressed in terms of its x, y, z components, and two scalars, R_1 and R_2, denoting the radii of the lower and upper bases.

ORNL-DWG 74-6766

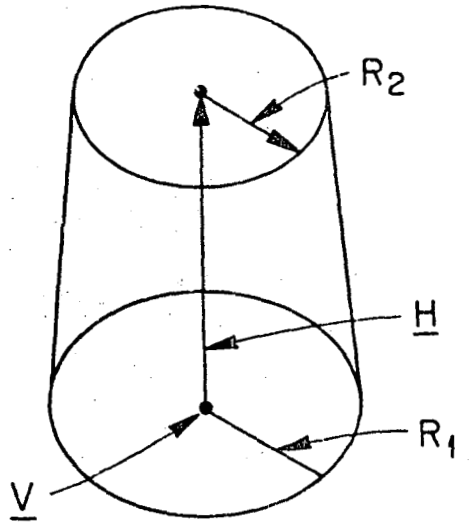


Fig. 4.13. Truncated Right Angle Cone (TRC).

f. Ellipsoid (ELL)

Specify two vertices, V_1 and V_2, denoting the coordinates of the foci and a scalar, R, denoting the length of the major axis.

ORNL-DWG 74-6767

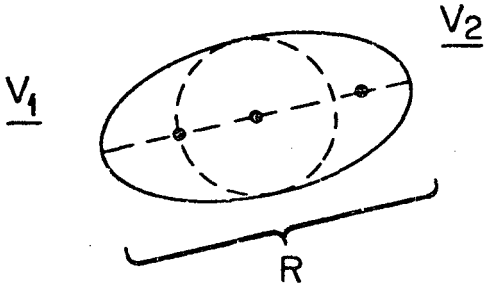


Fig. 4.14. Ellipsoid (ELL).

g. Wedge (WED) or (RAW)

Specify the vertex V at one of the corners by giving its (x, y, z) coordinates. Specify a set of three mutually perpendicular vectors, a_i, with a_1 and a_2 describing the two legs of the right triangle of the wedge. That is, the x, y, and z components of the height, width, and length vectors are given.

ORNL-DWG 74-6768

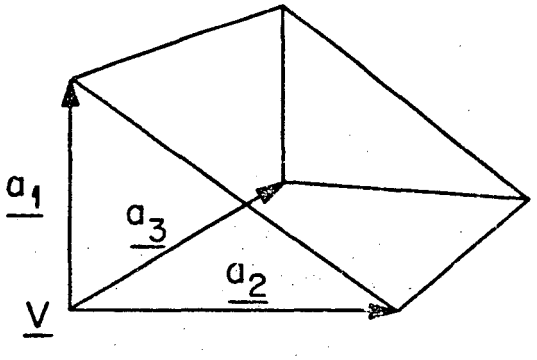


Fig. 4.15. Right Angle Wedge (WED).

h. Box (BØX)

Specify the vertex $\underline{V}$ at one of the corners by giving its (x,y,z) coordinates. Specify a set of three mutually perpendicular vectors, a_i, representing the height, width, and length of the box, respectively. That is, the x, y, and z components of the height, width, and length vectors are given.

ORNL-DWG 74-6769

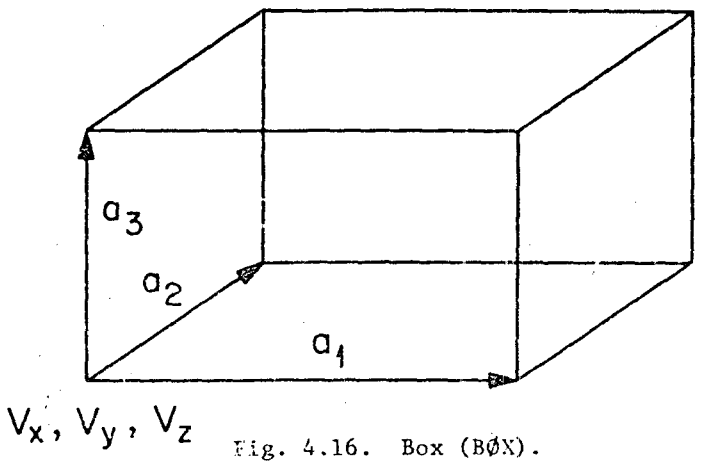


Fig. 4.16. Box (BØX).

i. Arbitrary Polyhedron (ARB)

Assign an index (1 to 8) to each vertex. For each vertex, give the x, y, z coordinates. Each of the six faces are then described by a four-digit number giving the indices of the four vertex points in that face. For each face these indices must be entered in either clockwise or counterclockwise order.

ORNL-DWG 74-6770

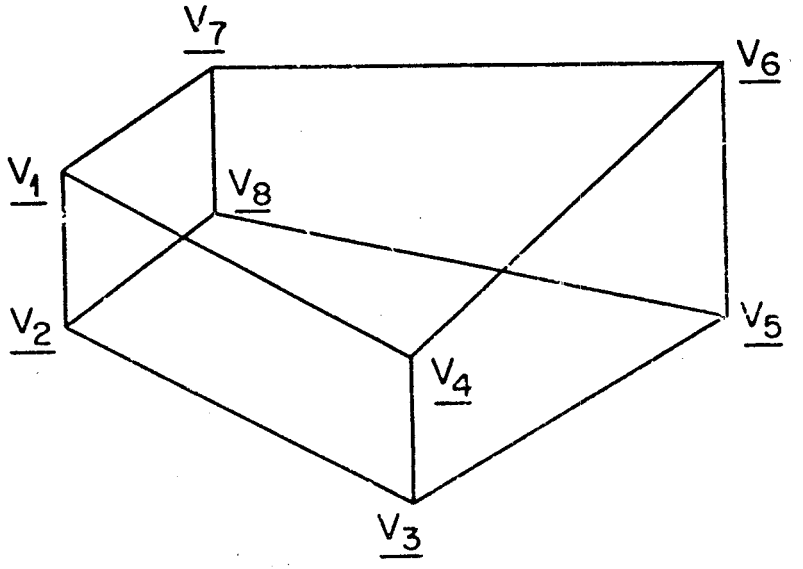


Fig. 4.17. Arbitrary Polyhedron (ARB).

Detailed input instructions are given in Section 4.3.

4.7.4. Subroutines

Subroutine G1 (S, MA, FFD, LØCREG, NUMBØD, IRØR, IR1, IR2)

G1 is the control routine for the combinatorial geometry. On one call, it calculates the distance traveled in the present zone, and the number IR of the next zone to be entered. Essentially, GG is called for each body adjacent to the present zone, calculating RIN and RØUT, the distances to entry and exit of the body along the trajectory. The next zone to be entered is determined by again calling GG to calculate RIN and RØUT for each body adjacent to the next possible zone. These next possible zones are determined by examining a list of all the previous zones entered on crossing this body. RIN and RØUT are checked against the input zone descriptions to determine the correct zone. If it is not found in the list of previous zones, all other zones are examined in a similar fashion, and when the correct zone is found, it is added to the list of previous zones for that body, if the storage allocated is not yet exhausted. If no more storage is available, future calls will search the list as it was adding no more to it but examining all other zones if it doesn't find the correct zone in the existing list. If the new zone is different from the old, G1 returns; otherwise G1 continues tracking until a different zone is encountered. One change added to the MØRSE version of G1 is that if the distance to the next boundary is greater than the distance to scattering, G1 returns without determining the next zone past the boundary, setting the flag MARKG in common ØRGI.

Called from: GØMST, EUCLID, MESH

Subroutines called: GG

Commons required: PAREM, GØMLØC, DBG, ØRGI, TAPE

Variables required:

XB(3)	- starting coordinates of present trajectory.
WB(3)	- direction cosines of trajectory.
IR	- present zone.
DIST	- present distance from XB(3).
DISTO	- distance from XB(3) to next scattering point.
NASC	- less than zero if this is a new trajectory.
KLØØP	- trajectory index.
PINF	- machine infinity.

MA, FPD, LØCREG,
NUMBØD, IRØR,
IR1, IR2 - locations in blank common used for variable dimensioning.

Variables changed:

KLØØP - trajectory index incremented if this is a new trajectory.

NASC - next body intersected by trajectory.

LSURF - surface of body NASC crossed at next intersection (negative if leaving and positive if entering NASC).

DIST - distance from XB(3) to next intersection or collision site.

MARK - set to 1 if distance to collision (DISTO) is less than distance to next intersection (otherwise 0).

S - distance traveled on this call to G1.

IRPRIM - zone to be entered on boundary crossing.

MA(INEXT) - new zone added to the list of next possible zones for body NBØ.

MA(INEX) - location in MA of next item in the next possible zone list for body NBØ (these lists leap-frog through the end of the MA array).

Significant internal variables:

NBØ - absolute value is body being considered while a negative or positive sign indicates that zone IR or IRP is outside or inside the body respectively.

RØUT - distance to exit of body NBØ calculated by GG.

RIN - distance to entry of body NBØ calculated by GG.

LRI - surface of body NBØ entered by trajectory.

LRØ - surface of body NBØ trajectory exits.

IRP - zone being considered as next zone.

Errors:

If message is "exit being called . . . next region not found" - NASC will be next body the ray will intersect (NBØ, N, NUM, ITYPE, SMIN, IRP, LØCAT all apply to maximum body number and have no significance).

Subroutine ALBERT (F, IERR)

ALBERT is called by subroutine GENI to process the arbitrary polyhedron (ARB) body data before storage in the FPD array. The ARB body data as read by GENI consists of the coordinates of all corners (eight for a six-sided figure), followed by a packed decimal number for each side indicating which corner points make up that side. ALBERT processes this data, returning a unit normal vector and a minimum distance to origin for each plane containing a side of the ARB. An ARB can have up to six sides. These unit vectors and distances then replace the original ARB data in the FPD array. The number of sides and a distance that is characteristic of the ARB's minimum dimension are also stored in the FPD array. This minimum distance is later used for round-off tests.

Called from: GENI

Subroutines called: none

Commons required: none

Variables required:

F - FPD array from GENI.

Variables changed:

F - FPD array changed to contain unit normals and distances to origin for each plane. Number of sides and a minimum distance are also stored.

Significant internal variables:

X(8,3) - coordinates of corner points.

IS(6,4) - indicates corners contained in each of the possible six surfaces.

V(3,4) - edge vectors for a given surface (an inner cross product then gives the normal vector).

Subroutine GENI (MA, FPD, LØCREG, NUMBØD, IRØR, MRIZ, MRCZ, MMIZ, MMCZ, KR1, KR2, I1, IØUT, IN)

Although subroutine JØMIN is used to read in all of the combinatorial geometry data except the region volumes, that data is put into the proper storage location in blank common by GENI. This is accomplished by having JØMIN write the body and zone data on a mass storage unit, so that GENI can retrieve that data. GENI assigns the data to blank common in the area set aside by JØMIN. For an arbitrary polyhedron body, GENI calls ALBERT to handle a portion of the data. In addition, GENI outputs the input information concerning the geometry data. GENI computes certain geometry dependent data needed during the random walk.

Called from: JØMIN

Subroutines called: ALBERT

Commons required: Blank, GØMLØC, PAREM

Variables required:

- I1 - input unit.
- IØUT - output unit.
- IN - bulk storage logical unit.

Variables input: Geometry data for zones and bodies is retrieved from mass storage unit IN.

Variables changed: MA(I), FPD(I), LØCREG(I), IRØR(I), MRCZ(I), MMCZ(I), KR2(I) - see Fig. 4.18 layout of combinatorial geometry storage in blank common.

Subroutine GG (LØCAT, MA, FPD)

GG is the workhorse of the combinatorial geometry, computing distances to intersections for all body types. It is called from G1 or LØØKZ to compute distance to entry and distance to exit to a body whose location in the MA array is flagged by the argument LØCAT. Each time a distance to entry, RIN, and a distance to exit, RØUT, are calculated for a body they are stored in the MA array together with LRI and LRØ, the indices of the entry and exit surfaces of the body. Also stored at this time is KLØØP, the particles trajectory index. On a subsequent call to GG for that body, KLØØP is checked against the earlier value, now LØØP. If they are the same, the old values of RIN, RØUT, LRI, and LRØ are retrieved so that GG can return immediately.

If it is necessary to compute a new trajectory, a different area of coding is entered for each body type to calculate RIN, RØUT, LRI and LRØ.

Called from: G1 and LØØKZ

Subroutines called: none

Commons required: PAREM, TAPE

Variables required:

- LØCAT - starting location in the MA array of integer data for the appropriate body.
- KLØØP - trajectory index.
- PINF - machine infinity (stored in RIN and RØUT when the trajectory misses the body).
- MA, FPD - locations in blank common required for variable dimensioning.

Variables changed:

- RIN - distance to entrance.
- RØUT - distance to exit.
- LRI - surface of entrance.
- LRØ - surface of exit.

Significant internal variables:

- L - same as LØCAT, starting location in MA array of body data for the appropriate body.
- K - starting location in FPD array of floating point data for the appropriate body.

Subroutine GTVLIN (FPD, MRIZ, VNØR, IVØPT, NIR, NIZ, I1, IO)

This routine is called by JØMIN to read in or to calculate the volume of each region in the geometry. The four options available are (1) to set each volume to 1., (2) to calculate the volumes for concentric spheres, (3) to calculate the volumes for slabs (not coded at present), or (4) to read in the volumes for each region from cards. The volumes are stored in blank common and are only used by the track length and collision density estimators.

Called from: JØMIN

Subroutines called: none

Commons required: none

Variables required:

FPD	- array containing zone description data.
MRIZ	- array which related MØRSE region to input zones.
VNØR	- array containing the MØRSE volumes.
IVØPT	- options for determining volumes.
NIR	- number of regions.
NIZ	- number of zones.
I1, IO	- input and output logical units.

Variables changed:

Variables in blank common starting at KVØL.

Subroutine JØMIN (NADD, I1, IO)

The input of the geometry data is controlled by the JØMIN subroutine, which performs the following tasks:

- Reads all geometry input data except the region volumes.
- Writes the body and zone data on a mass storage unit (IØUT=16).
- Determines the length of all geometry arrays.
- Calculates the beginning location in blank common of geometry arrays.
- Initializes geometry arrays.
- Calls the GTVLIN subroutine which returns region volumes.

Since combinatorial geometry input data is dynamically allocated to conserve storage area, it is stored temporarily on a mass storage device. This allows the core storage requirements to be determined. Hence, much of the coding in JØMIN is similar to GENI, which reads the data on the mass storage device and puts it into blank common.

Called from: INPUT1

Subroutines called: GENI, GTVLIN

Commons required: Blank, GØMLØC, PAREM, TAPE

Variables required:

All variables in GØMLØC, I1, IO.

Variables changed:

All variables in GØMLØC, IVØPT, IDBG, MRIZ(I), MMIZ(I).

Important internal variables:

IØUT - mass storage unit.

NAZT - total number of adjacent zones summed over all zones.

Subroutine LØØKZ (X, Y, Z, MA, FPD, LØCREG, NUMBØD, IRØR, NSØR)

The purpose of this routine is to return the combinatorial geometry zone of point (X, Y, Z) so that tracking can be initialized. The coding has been borrowed from the second half of subroutine G1 and adapted to determine the zone of a source particle. For efficiency LØØKZ builds a list of possible source zones to search on future calls. If the region is not found on this list, all other zones are examined and upon determining the new source zone, it too is added to the list. Notice that the starting direction cosines (.8, .6, 0.0) are assumed in LØØKZ, but may be changed elsewhere.

Routines called: GG

Commons required: ØRGI, PAREM, GØMLØC, and DBG.

Variables required:

X, Y, Z - coordinates for which a zone is desired.

MA, FPD, LØCREG, NUMBØD, IRØR, NSØR - locations in blank common used for variable dimensioning.

Variables changed:

KLØØP - trajectory index is incremented.

NMED - common ØRGI variable set to correct zone number.

NSØR(INEXT) - new zone added to list of possible source zones.

Significant internal variables:

WB(3) - set to .8, .6, 0.0 so that LØØKZ need not be called with a direction.

Subroutine NØRML (MA, FPD, LØCREG, NUMBØD)

The purpose of subroutine NØRML is to return a unit vector to a combinatorial geometry body NASC at point $\vec{XB}$ + DIST * $\vec{WB}$ which must be on the surface LSURF of body NASC. This unit vector is useful either for albedo scattering or boundary crossing flux estimates. The sign of the unit vector is chosen so that $\vec{WB} \cdot \hat{n}$ is negative meaning that the unit vector will point against the particle direction.

Called from: GØMST

Commons required: NØRMAL, GØMLØC, PAREM

Variables required:

NASC - body number particle is on.

LSURF - surface of body NASC.

XB(3), WB(3), DIST - trajectory parameters.

MA, FPD, LØCREG, NUMBØD - locations in blank common used for variable dimensioning.

Variables changed:

UN(3) - direction cosines of unit vector stored in common NØRMAL.

Significant internal variables:

XP(3) - set to $\vec{XB}$ + DIST * $\vec{WB}$.

X(3) - usually used to relate the intersection point to a body centered coordinate system.

H(3) - usually used as a body orientation.

L - body location in MA array.

K - body location in FPD array.

Subroutine PR (K)

Subroutine PR is called from various locations in the combinatorial geometry package (GENI, G1, LØØKZ and NØRML) whenever intermediate or debugging output is required. The amount of geometry data which is output depends on the value of the argument. If this argument is 1 or 8, all of the geometry data in blank common is printed, otherwise only selected variables are output. By comparing the argument value given in the output with the source listing, the geometry data at a given time in the execution can be determined. The call to PR is initiated by setting the IDBG variable to a nonzero value. Because of the large amount of output generated, this option should not be used during a normal execution.

Called from: GENI, G1, LØØKZ, NØRML

Subroutine called: none

Commons required: blank, PAREM, GØMLØC, DBG, TAPE

Variables required:

K - a signal indicating how much data to print
= 1 or 8-print all geometry data in blank common.
Otherwise, print selected variables only.

Starting Location	Information	Size
NGEØM=NADD	Length of geometry array	1
KMA	MA Integer array	LTMA
KFPD	FPD Floating point array	LFPD
KLCR	LØCREG Indices to correlate MA array data with code zone data	NUMR
KNBD	NUMBØD Number of bodies for each code zone	NUMR
KIØR	IRØR Indices to correlate input zone to code zone	NUMR
KRIZ	MRIZ Indices to correlate MØRSE region to input zone	IRTRU
KRCZ	MRCZ Indices to correlate MØRSE region to code zone	NUMR
KMIZ	NMIZ Indices to correlate MØRSE media to input zone	IRTRU
KMCZ	NMCZ Indices to correlate MØRSE media to code zone	NUMR
KKR1	KR1 Indices to correlate first code zone to input zones	IRTRU
KKR2	KR2 Indices to correlate last code zone to input zone	IRTRU
KNSR	NSØR Indices of code zones in which source particles have been found	NUMR
KVØL	VNØR Volume of each MØRSE region	NIR
NGLAST		

Fig. 4.18. Layout of Combinatorial Geometry Data in Blank Common.

Position in Blank Common	Information Stored	Size	Description
KFPD	RIN for body 1	1	Path length data for last trajectory in body 1.
KFPD + 1	RØUT for body 1	1	
KFPD + 2	First 6 words of real data for body 1	6	Read from first card of body 1 card set.
KFPD + 8	Remaining words of real data for body 1	N_1	N_1 depends on body type. See Table 4.3.
KFPD + 8 + N_1	RIN for body 2	1	
	RØUT for body 2	1	Same information as above, but for body 2.
	Remaining data for body 2 (N_2 words)	N_2	
	Repeat for NUMB bodies.		

NOTE: 8 words are set aside at the end of the FPD array, but are not used.

Fig. 4.19. Detailed Layout of the FPD Array in Blank Common.

Position in Blank Common	Information Stored	Size	Description
KMA	Body number 1	7	Body number 1 data (each body requires 7 words of information).
KMA + 1	LØØP for body number 1		
KMA + 2	Body type (ITYPE)		
KMA + 3	LRI		
KMA + 4	LRØ		
KMA + 5			
KMA + 6	Beginning location in FPD body data		
KMA + 7	Body number 2	7	Same information and order as for body number 1.
KMA + 14	↓		
KMA+(L-1)*7	Body number (L)	7	Body number L is the last body.

Fig. 4.20. Detailed Layout of the MA Array in Blank Common.

Position in Blank Common	Information Stored	Size	Description
KMA + L*7	Zone number 1	1	Beginning of code zone information
	Number of first body in this zone	4	Beginning of information about bodies defining code zone 1. Integer data location is given by: 7*(Body number) - 6.
	Location of integer data for this body		
	First zone to search upon exiting this body		
	Location of next zone to be searched		
	Data on second body in this zone	4	The last two words in each set of body data initiate the "leap frog" process by which the code stores possible zones which can be entered upon exiting this body in that particular zone. These zones are checked by the code when the next zone entered is being determined. If the next zone is not located from this stored data, all zones are searched.
	Data on last body in the zone	4	

Fig. 4.20. Detailed Layout of the MA Array in Blank Common (continued)

Position in Blank Common	Information Stored	Size	Description
	Zone number 2	1	
	Body information		Same information as above except for code zone number 2.
	Zone data of last zone		Code zone information about the last zone input on cards.
KMA + LDATA	Code search information KMA+LTMA-1	2*NAZT	Storage set aside for determining the zone to be searched and where the next zone number is located.

Fig. 4.20. Detailed Layout of the MA Array in Blank Common (continued)

Table 4.23. Definitions of Variables in Common GØMLØC

Variable	Definition
KMA	Starting location for the array MA containing integer data for each code zone.
KFPD	Starting location for the array FPD containing real data for each code zone.
KLCR	Starting location for the array LØCREG(I) which contains the starting location in the MA array for the I^{th} code zone data.
KNBD	Starting location for the array NUMBØD(I) which contains the number of bodies for the I^{th} code zone.
KIØR	Starting location for the array IRØR(I) which contains the index of the corresponding input zone for the I^{th} code zone.
KRIZ	Starting location for the array MIRZ(I) which contains the index of the MØRSE region corresponding to the I^{th} geometry input zone.
KRCZ	Starting location for the array MRCZ(I) which contains the index of the MØRSE region corresponding to the I^{th} geometry code zone.
KMIZ	Starting location for the array MMIZ(I) which contains the index of the MØRSE media corresponding to the I^{th} geometry input zone.
KMCZ	Starting location for the array MMCZ(I) which contains the index of the MØRSE media corresponding to the I^{th} code zone.
KKR1	Starting location for the array KR1(L) contains the first code zone which was made from the L^{th} input zone.
KKR2	Starting location for the array KR2(L) contains the last code zone which was made from the L^{th} input zone.
KNSR	Starting location for array NSØR which contains the code zones in which source particles have been found.
KVØL	Starting location for the array VNØR(I) which contains the volume for MØRSE region (I).

Table 4.23 (Cont'd.)

Variable	Definition
NADD	Starting location for the geometry data length and changed in JØMIN to the total number of words required for geometry data.
LDATA	Length of the integer data in the MA array excluding the words set aside for zone search information.
LTMA	Total length of the MA array.
LFPD	Length of the FPD array.
NUMR	Number of code produced zones.
IRIRU	Number of input zones.
NUMB	Number of bodies.
NIR	Number of MØRSE geometry regions.

Table 4.24. Definitions of Variables in Common PAREM† as Found in Combinatorial Geometry

Variable	Definition
XB(3)	Coordinates of the starting point of the present path. Changed in EUCLID, GOMST, and LOOKZ.
WB(3)	Direction cosines of particle trajectory. Equal to U, V, and W. Changed in EUCLID, GOMST, and LOOKZ.
WP(3)	Temporary storage of WB(3).
XP(3)	Temporary storage of XB(3). Changed in NORML.
RIN	Distance to entry calculated in GG.
RØUT	Distance to exit calculated in GG.
PINF	Machine infinity. (1.0E + 20)
DIST	Distance from XB(3) to present point.
IR	Combinatorial zone of present particle position.
IDBG	Set non-zero to initialize a debug printout.
IRPRIM	Next region to be entered after a call of G1.
NASC	Body number of last calculated intersection. Set negative to indicate source or collision point not on a body surface.
LSURF	Surface of body NASC where intersection occurred. Positive if particle is entering the body and negative when exiting.
N3Ø	Body number and a sign used to define zones. Input in zone description as positive when zone is contained in body and as negative if zone is outside body.
LRI	Entry surface calculated in GG.

†The order of the variables in PAREM in versions other than IBM-360 is XB, WB, IR, WP, XP, IDBG, IRPRIM, NASC, LSURF, NBØ, LRI, LRØ, RIN, RØUT, KLØØP, LØØP, ITYPE, PINF, NØA, DIST. Because the IBM-360 version is double precision, it required reordering the variables.

Table 4.24 (Cont'd.)

Variable	Definition
LRØ	Exit surface calculated in GG.
KLØØP	Trajectory index of present path incremented in G1.
LØØP	Index of last trajectory calculated for body NBØ. If LØØP is equal to KLØØP, GG returns immediately with old values in RIN, RØUT, LRI, and LRØ.
ITYPE	Body type of body NBØ (indicates BØX, SPH, etc.).
NØA	Not used.

Table 4.25. Definitions of Variables in Common ØRCI*

Variable	Definition
DISTO	Distance from point XB(3) to next scattering point. Used in G1 to avoid calculating the next zone if a scattering event occurs before the intersection.
MARK	Set 1 in G1 if trajectory end point is reached before next intersection. Otherwise set to 0.
NMEDG	Zone number IR from a LØØKZ call. Stored in BLZNI by MSØUR.

* Note: Variable names are not the same in all routines. Also, on non-IBM-360 machines, the order of the variables is MARK, DISTO, NMEDG. Reordering resulted from the conversion of the IBM-360 version to double precision.

4.8. DIAGNOSTIC MODULE

4.8.1. Introduction

Frequently in debugging a problem or in trying to gain further insight into the physics of a problem, it is desirable to dump the contents of certain labelled commons or parts of blank common. This module of MØRSE makes it possible to print out in a readable format the values of these variables.

The key routine in the IBM-360 version is subroutine HELPER* which prints out, in decimal form, any part of a single-precision (4-byte word) array. This routine, along with two machine-language (IBM-360 series) routines, decides whether a number is an integer or a floating point number and converts to EBCDIC accordingly. It also recognizes the "junk" word (48484848_{16}) and outputs the string "NØT USED" in its place. This feature is included because it is not always feasible to depend on the core being zeroed or filled with any particular constant. Selected portions of core are therefore filled with this word, which was selected because it is essentially the same number when treated as an integer or as a floating point number. The features of this routine are not available on systems other than the IBM-360.

A more inclusive dump may be obtained with subroutine HELP which outputs, on request, selected portions of blank common and commons APØLLØ, FISBNK, NUTRØN, and USER.

In addition to these diagnostic aids there are numerous error messages printed from the routines. These messages are explained in Tables 4.26 and 4.27.

---

*HELPER is a slight revision of TDUMP.[16]

Table 4.26. Diagnostic Messages From Input Module in MØRSE

Subroutine Printing Message	Message	Meaning
INPUT2	'YOU CANT WIN THEM ALL'	Dimension of blank common is too small - increase the dimension in MAIN.
XSEC	'MEDIA IN INPUT = ____ NMED IN XSEC = ____ NMGP = ____, NGP = ___ NMTG = ____, NTG = ___	Either the number of media or the number of groups is specified differently in MØRSE and in MØRSEC input.
XSEC	'TEMPORARY CROSS-SECTION STORAGE EXCEEDS BLANK COMMON BY xxx'	Dimension of blank common is too small to allow storage of temporary cross sections. Increase the dimension of blank common in MAIN by at least xxx.
XSEC & XSTAPE	'PERMANENT CROSS-SECTION STORAGE EXCEEDS BLANK COMMON BY xxx'	Same as above except this applies to permanent cross section storage.
JNPUT	'***JOB TERMINATED BECAUSE FIRST MOMENT WAS OUTSIDE RANGE, SEE PRINTOUT ABOVE FOR GROUPS**, ____ BAD MOMENTS WERE FOUND'	There is some error in the user's cross section data.
JNPUT	'********** IN MIXTURE ____, ELEMENT ____ WAS MIXED IN MEDIUM ____ THE RESTORED CROSS SECTIONS DESTROYED THE CROSS SECTIONS BEING PICKED UP. REORDER THE MIXING OPERATIONS OR THE ORDER OF ELEMENTS AS INPUT	Self-explanatory. Note: To date there have been no such occurrences.

Table 4.26 (Cont'd.)

Subroutine Printing Message	Message	Meaning
ANGLES	'FIRST MOMENT REJECTED. M(1) = ____, IN = ____, OUT = ____, MIXTURE = ____.'	Error is called. IN is the group doing scattering; ØUT is the group scattered to; MIXTURE is the media number.
BADMØM	'____MOMENT (____) IS BAD, OUTPUT FROM BADMOM' 'MUBOT = ____MU(____) = ____ MUTOP = ____' 'VARBOT = ____VAR(___) = ____ VARTOP = ____' 'MOMBOT = ____MOM(____) = ____ MOMTOP = ____ RANGE = ____ ERROR = ____' 'FBOT = ____F(____) = ____ FTOP = ____ RANGE = ____ ERROR = ____'	This output results from bad moments with the ___BOT and ___TOP being the allowable range.
FIND	'----FIND ROOTS OF Q (___)EXTEND BEYOND +1' '----FIND ROOTS OF Q (___)EXTEND BEYOND -1'	Printed only if IPUN > 0.
GTNDSK	'NOT ENOUGH CORE ALLOTTED, NEED ___'	Dimension in blank common is not large enough as set in MAIN.
GTSCT	'BLANK COMMON EXCEEDED IN READING O6R CROSS SECTIONS ___ MORE CELLS NEEDED'	Same as above.

Table 4.26 (Cont'd.)

Subroutine Printing Message	Message	Meaning
	'06R CROSS SECTIONS DO NOT COVER ENERGY RANGE REQUIRED BY MORSE GROUP IGQPT.'	Either IGQPT is specified wrong or the cross sections cover the wrong energy range.
READSG	'*** CARD OUT OF ORDER ***' 'CARD ___ WAS READ WHEN CARD ___ WAS EXPECTED. CHECK CROSS SECTION CARD SEQUENCE IN ELEMENT ___ COEFFICIENT ___'	Cards are not numbered sequentially - code will continue using them in the order they appeared but warns the user to check his card order.
READSG	'ONLY ___ CROSS SECTIONS WERE USED FROM THE FOLLOWING CARD'	There is a probable error in the number of cross sections on the last card for the cross sections of a given element. Too many numbers present.
READSG	'ELEMENT ___ CANNOT BE FOUND'	The requested element is not present on IXTAPE. Check the ID's on input and on tape.
RESTØR	'ELEMENT ___ CANNOT BE FOUND'	Same as above.
SCØRIN	'INSUFFICIENT ROOM FOR DETECTOR ARRAYS. ND = ___, NE = ___, NT = ___, NA = ___, LMAX = ___, NLAST = ___, NLFT = ___'	Dimension in blank common is not large enough to hold the analysis data. Increase dimension of blank common in MAIN.

Table 4.26 (Cont'd.)

Subroutine Printing Message	Message	Meaning
	'** ERROR IN NNE, NE OR THE GROUP NUMBERS -- PRINT IS OF MODIFIED VALUES'	Either IB(NNE) does not equal NGPQTN or IB(NE) does not equal the maximum number of groups. The routine modifies the user's data to correct this. User should change his IB array in his input.
JØMIN	'TYPE = ___ DOES NOT EQUAL ANY OF THE FOLLOWING ___ ___ ___'	The body type given does not exist in the code.
GENI	'ITYPE = ___ DOES NOT EQUAL ANY OF THE FOLLOWING ___ ___ ___'	The body type given does not exist in the code.
ALBERT	'ERROR IN SIDE DESCRIPTION ___ ___ ___'	There are less than 3 points that describe this side - i.e., the user has defined a point or a line rather than a surface.
	'ERROR IN FACE DESCRIPTION ___'	The points describing this face are not coplanar.
GTVLIN	'********* ERROR IN VOLUME CALCULATION ********* JF = ___ NIR = ___'	The number of regions for which volumes were calculated does not equal the number of regions in the geometry.

Table 4.27. Diagnostic Messages From Other Modules of MØRSE

Subroutine Printing Message	Message	Meaning
FBANK	'WARNING***NO ROOM IN BANK FOR SECONDARIES***'	Maximum number of particles have been generated and the bank is full. No more fission particles will be generated.
GSTØRE	'WARNING***NO ROOM IN BANK FOR SECONDARIES***'	Maximum number of particles have been generated and the bank is full. No more gamma rays will be generated until bank decreases.
FSØUR	'NO FISSIONS GENERATED IN LAST BATCH, ___ BATCHES COMPLETED'	Problem continues for a fixed source problem. For a criticality problem the neutron population has died away.
MØRSE	'NREG = ___ MXREG = ___ MXREG ON CARD I MUST BE GE TO THE NUMBER OF REGIONS DESCRIBED IN GEOMETRY INPUT'	Self explanatory - change your input data.
CØLISN	'IN GRP = ___, OUT GRP = ___ J = ___ PROB = ___ RAND = ___'	All the scattering angle probabilities are 0 or negative for this group. An index is wrong or the data on the angle of scattering has been destroyed.
PTHETA	'ERROR in PTHETA ___ ISTAT = ___ NCOEF = ___'	The user has called PTHETA without saving Legendre coefficients - ISTAT must be non zero.

Table 4.27 (Cont'd.)

Subroutine Printing Message	Message	Meaning
GØMST	'NAME = ___ NMED = ___ NREG = ___ X = ___ Y = ___ Z = ___'	An error has been encountered in the geometry tracking for this particle. There should be messages before and after this giving more information. G1 will print a message and PR will be called here. Particle will be treated as an escape and calculation will continue.
EUCLID	'IRPRIM = ___ IN EUCLID'	An error has occurred in calculating the number of mean free paths to the detector. There will be an error message preceding this which will come from G1. Program allows 5 such errors before termination. Particle will be treated as an escape and will make no contribution to estimate.
LØØKZ	'IR = ___ XB = ___ ___, WB = ___ ___, DIST = ___' 'MA ARRAY' Entire MA array printed 'FPD ARRAY' Entire FPD array printed '**** EXIT BEING CALLED FROM LOOKZ'	Zone has not been found. Check your zone specifications.

Table 4.27 (Cont'd.)

Subroutine Printing Message	Message	Meaning
NØRML	'INVALID REGION OR BODY IN NORMAL. IR = ___ NASC = ___'	NASC isn't in region IR Check to see if IR or NASC has been overwritten.
GG	'IN GG ITYPE = ___ IR = ___ NBO = ___'	ITYPE is not one of the body types (1 - 9) allowed. User has usually overstored on MA array.
G1	'NO VALID DISTANCE IN G1, ___'	G1 could not determine the next body that the ray will intersect. There is a probable error in user's geometry specifications, or he may have written over his geometry data.
	'********** GEOMETRY SEARCH ARRAY FULL **********'	In order to save some computer time, user may want to increase the values of NAZ on his zone specification cards. Only harm done is increase in computer time which is often insignificant.
	'IR = ___XB = ___WB = ___ DIST = ___'	G1 could not find the next region that the particle would enter. It checked all regions and is saying that the particle won't enter any of them.
	'MA ARRAY' Entire MA array printed	
	'FPD ARRAY' Entire FPD array printed	

Table 4.27 (Cont'd.)

Subroutine Printing Message	Message	Meaning
G1	'EXIT BEING CALLED FROM G1, NEXT REGION NOT FOUND.'	Probable error in zone specifications. Subroutine PR is called to print out values from the geometry commons GØMLØC, DBG, and PAREM. Among the variables printed the following apply to the maximum body number and therefore have no significance - NBØ, N, NUM, ITYPE, SMIN, IRP and LØCAT. The important variable is NASC which is the number of the next body that the ray will intersect. That is, the code can not find what region this body is in. Check your zone specifications for this body.
NØRML	'ROUND-OFF ERROR IN NORMAL NBO=_ LSURF=__XP=__ __ __ '	Self-explanatory. Applies only to ELL and BØX.

4.8.2. Subroutines

Subroutine BNKHLP (NAME)*

This routine outputs (one particle to a line) all of the particle bank and, if used, all of the fission bank. If identical lines are encountered, it prints a message giving the number of identical lines. The last line is always printed.

Called from:

HELP - when index IGXBP < 0.

Subroutines called:

ICØMPA (A,B,N) (library function at Oak Ridge National Laboratory - compares, bit by bit, N bytes of locations A and B; returns zero if A and B are identical.)

Commons required:

Blank, APØLLØ, FISBNK

Variables required:

NSIGL - location in blank common of cell zero of the particle bank.

NMØST - maximum number of particles allowed for in the bank(s).

IO - logical unit for output.

MFISTP - index indicating that fissions are to be considered if > 0.

NFISBN - location in blank common of cell zero of the fission bank.

---

*This routine is a dummy on machines other than the IBM-360.

Subroutine HELP (ICALL, INUMP, ILABP, IGXBP, IUSRP)

This routine is used to output values of selected variables used by the code, at any desired point in the solution of the problem. It will provide, with setting of the proper switch, prints of:

1) blank common from cell one up to the geometry data storage,
2) first and last eight words of geometry and cross-section data storage areas,
3) first and last 12 words of the neutron bank, or the entire neutron and fission (if used) banks,
4) all the user area in blank common (beyond the neutron and fission banks), and
5) labelled commons APØLLØ, FISBNK, NUTRØN, and USER.

HELP has been found useful to the writers of the code in debugging. For this purpose, temporary calls are inserted at points of interest. As the code stands now, calls are made in MØRSE at a few points in the code that will not be reached unless an error occurs.

Called from: MØRSE, FBANK, FPRØB, GPRØB

Subroutines called:

HELPER*

BNKHLP* - prints all of the neutron bank and all of the fission bank if it is being used.

Function used: LØC

Commons required: Blank, NUTRØN, FISBNK, APØLLØ, USER

Variables required:

ICALL - 4 EBCDIC characters representing location of call.

INUMP - > 0 for print of blank common.

ILABP - > 0 for print of labelled commons.

IGXBP - > 0 for print of first and last 8 cells of geometry and cross-section storage, and the first and last 12 cells of the bank;
< 0 for above print of geometry and cross section and also to call BNKHLP for complete print of the neutron and fission (if used) banks.

---

*This routine is a dummy except on the IBM-360 system.

IUSRP - > 0 for print of user area in blank common.

NMTG - total number of energy groups.

LØCWTS - location of cell zero of weight standards arrays.

MGPREG - product of number of groups and regions for weight standards.

LØCFWL - location of cell zero of FWLØ array.

MXREG - number of regions for weight standards.

LØCEPR - location of cell zero of energy group bias array, (= 0 if energy group bias not being used).

LØCNSC - location of cell zero of scattering counter arrays.

MEDIA - number of media in cross sections.

LØCFSN - location of cell zero of FISH array.

NGEØM - location of cell one of geometry data storage.

NSIGL - location of last cell in permanent cross-section storage.

NLAST - last cell used by neutron or fission bank.

NLEFT - number of cells available to user beyond banks.

Subroutine HELPER (A, INIT, NLAST, NAME, IO)*

HELPER enables the user to output, in decimal form, any part of a single-precision (4-byte word) array at any point in the program. The user need not know whether the numbers are integer or floating point. Numbers that can be translated as integers in the range $\pm 16^6 (\pm 16777216)$ will be printed as such; floating numbers are handled correctly between $\pm 16^{-64} (\sim 10^{-76})$ and $\pm 16^{63} (\sim 10^{75})$. If the junk word (48484848_{16}) is encountered, "NØT USED" is printed. Numbers are printed eight to a line in an E11.5 or I11 format and identical lines are replaced by a "REPEATING LINE PATTERN" message (except that the last line of an array output is always printed).

Called from: HELP, XSCHLP

Subroutines called:

SUBRT

ICØMPA - (library function at Oak Ridge National Laboratory; see BNKHLP writeup).

Variables required:

A - first word of array of interest.

INIT - first 4-byte word of array A to be output.

NLAST - last 4-byte word of array to be output.

NAME - 4 hollerith characters to be used as a label.

IO - output unit.

---

*This routine is a dummy on machines other than the IBM-360.

Subroutine SUBRT (A, N, A1)*

SUBRT is an assembly language routine called by HELPER to perform conversion of a 4-byte computer word to a string of hollerith characters. It tests for unused elements (48484848_{16}) returning the string "NØT USED", decides whether the number is an integer or floating point, converts the number into hollerith if floating point, and calls INTBCD if integer. INTBCD is called to convert all numbers it receives as integers into hollerith and passes control back to SUBRT.

Called from: HELPER

Routines called:

INTBCD - library subroutine at Oak Ridge National Laboratory; converts a 4-byte integer to an EBCDIC string.

Variables required:

A - 4-byte word to be converted.

N - format size (HELPER calls with N = 11 resulting in I11 and 1PE11.5 formats).

Variables changed:

A1 - first word of 12-byte array for storage of hollerith string.

*This routine is not used on machines other than the IBM-360.

Subroutine XSCHLP (IBCDUM, NAME)

This routine outputs in decimal or integer form the contents of the commons used in the cross-section module, as well as the contents of the various cross-section arrays in blank common. (See Table 4.16 for layout of the cross-section area.) This subroutine may be called from any location. Versions other than the IBM-360 print only the values in LØCSIG common.

Called from: READSG, PTHETA, XSEC, and ANGLE (just before error calls).

Subroutines called: HELPER (by IBM-360 version).

Functions used: LØC (by IBM-360 version).

Commons required: Blank, LØCSIG, MEANS, MØMENT, QAL, RESULT.

Variables required:

IBCDUM - contents of blank common are printed if > 0.

NAME - a four-character word to indicate the calling program.

4.9. HARDWARE AND SOFTWARE REQUIREMENTS

4.9.1. Hardware Requirements

Three versions of MØRSE are available; one each for the IBM-360 system, the CDC-6600 system, and the UNIVAC-1108 system. It has been run on the IBM-360/75,/91 and /195, on the CDC-6600 and on the UNIVAC-1108. As previously mentioned in Section 4.4 the main memory storage requirement in bytes is of the order of 155,000+4*(blank common size in words) not including system library routines. The combinatorial geometry input routines use temporary storage on logical unit 16 which should preferably be a disk. The amount of storage needed on the disk depends on the number of bodies and zones in the system since the actual geometry input data is written out on unit 16. The standard input and output unit numbers are specified by the user in the main program. No other tapes or disk space are required unless the user has his cross section data on tape or disk or is writing a collision tape in which case he specifies the logical unit numbers in his input data. MØRSE also samples the clock to determine the c.p.u. time used.

4.9.2. Library Routines and Functions

The library routines used by the MØRSE system include the standard I/Ø routines and the following mathematical functions:

ABS, IABS, and DABS
EXP
MAXO
MØD and AMØD
SIGN and DSIGN
SQRT and DSQRT

In addition to these routines, there are certain other subroutines and functions used which are library routines at the particular installations where the different versions of MØRSE were developed. These routines whose descriptions follow are not provided with MØRSE.

Subroutine or Function	Called From	System Using It
LØC	MAIN, XSCHLP, HELP	IBM-360
INTØBC	DATE	IBM-360
INTBCD	DATE, TIMER, SUBRT, NRUN	IBM-360
ICØMPA	INPUT1, BNKHLP, HELPER	IBM-360
MØDEL	INPUT1	IBM-360
IDAY	IWEEK	IBM-360
ICLØCK	TIMER, MØRSE	IBM-360
INSERT	TIMER, NRUN	IBM-360
ERTRAN	DATE	UNIVAC-1108
TICKER	CPUTIM	UNIVAC-1108
SECØND	CPUTIM	CDC-6600

There are several uses of these library subroutines. One is to provide the time, day of the week, and year that the job is being executed. A second use, provided by Subroutine TIMER, is in determining the amount of c.p.u. time used per batch and for input and output. To obviate several of these library subroutines, dummy subroutines TIMER and DATE may be used. A third use is in the diagnostic module. The absolute location of variables in commons, the determination of a repeating array, a "not used" feature, and an integer or floating point

output are the features of the diagnostic module that require these special routines. If similar routines are not available, other user-written routines can be supplied for XSCHLP, BNKHLP, and HELP.

Several other uses of these routines are made, but they are relatively unimportant. MODEL is used to scale MAXTIM on the IBM-360 system, depending on the machine on which the job is being executed. ICØMPA is used by INPUT1 to terminate a job when a non-blank or alphanumeric character appears in the first column of Card A. While none of these features are necessary to the operation of MØRSE, they have proven to be quite useful.

Function ICLØCK

This function is used in the following manner:

I = ICLØCK (0)

The function returns the reading of the computer timer with the value, in hundredths (.01) of seconds, to be stored in an integer*4 variable. For compatibility with older systems, the function may be called by the name ICLØCKF. For example: I = ICLØCKF (0). The argument of the function is not used, so 0 is suggested for use.

The difference between two successive calls of ICLØCK is the time in hundredths of seconds between the two calls.

Example:

```
      I = ICLØCK (0)
computation portion of program which takes 42.75 seconds of CPU time
      J = (ICLØCK (0)-I)/100
      PRINT 1, J
    1 FORMAT (8H_TIME_=_,I5,20H_SECONDS_OF_CPU_TIME)
    TIME =    42 SECONDS OF CPU TIME would be printed out.
```

Function ICØMPA

This function is used in the following manner:

I = ICØMPA (ARG1, ARG2, N)

The result of the function, stored in the integer*4 variable I, is -1, 0, or 1. The function compares ARG1 and ARG2. The number of bytes to be compared is specified by the integer*4 value N where N may be 1 through 256. If, for example, N is 1 then the first byte of ARG1 is compared with the first byte of ARG2. If N is 4 then the first four bytes of ARG1 are compared with the first 4 bytes of ARG2.

This function is particularly useful in comparing alphanumeric characters. This avoids any overflow or underflow interrupts which might occur in the arithmetic subtraction of two variables containing alphanumeric data. Also this function allows only the number of characters of interest to be compared which eliminates the need for blank filling of alphanumeric data in some instances.

Comparison starts at the locations specified by ARG1 and ARG2 and proceeds from left to right, bit by bit. The result returned by the function will be 0 if the specified number of bytes of ARG1 are the same as those of ARG2; the result, 1, will be returned if, going from left to right, a bit in ARG1 is 1 and the corresponding bit in ARG2 is 0; the result, -1, will be returned if, going from left to right, a bit in ARG1 is 0 and the corresponding bit in ARG2 is 1.

If N is less than or equal to 0, the value returned by the function is 0. If N is greater than 256, the result will be meaningless. The function, ICØMPA, is also available on the library under the name ICØMPARE, thus I = ICØMPARE (ARG1, ARG2, N).

Example: Test the first two characters of Q to see if they are the letters AB. If they are, branch to statement 20.

```
     READ 1, Q
   1 FØRMAT (A4)
     IF(ICØMPA (Q, 2HAB, 2)  .EQ.  0) GØ TØ 20
```

Example: Test the 27th character on a card to see if it is the letter X. If it is, branch to statement 40.

```
      LØGICAL*1 Q(80)
      READ 1, Q
    1 FØRMAT (80A1)
      IF(ICØMPA (Q(27) , 'X', 1)  .EQ. 0) GØ TØ 40
```

Example: Ten input cards with identifying characters punched in column 1 and 2 and data in columns 3 through 8 are to be read into the computer and sorted in ascending order of the identifying characters.

```
      REAL*8 A(10), TEMP
      READ 1, A
    1 FØRMAT (A8)
    2 DO 3 I = 1, 9
      IF(ICØMPA (A(I), A(I+1), 2) .LE. 0) GØ TØ 3
      TEMP = A(I)
      A(I) = A(I+1)
      A(I+1) = TEMP
      GØ TØ 2
    3 CØNTINUE
      END
```

Subroutine IDAY

This subroutine is called in the following manner:

```
CALL IDAY (WHEN)
```

The subroutine returns the date as 8 EBCDIC characters consisting of the month, day and year separated by minus signs, for example, 05-20-55. The eight byte date will be placed in the location designated by the argument. The argument may be one word of type real*8 or a dimensioned variable of at least two words of type real*4 or integer*4. The date may then be printed using an A8 format (with a real*8 argument) or a 2A4 format (with two real*4 or integer*4 words).

Example: Program run on May 20, 1968.

```
     REAL*8 TØDAY
     CALL IDAY (TØDAY)
     PRINT 2, TØDAY
   2 FØRMAT (10H_TØDAY_IS_,A8)
```

TODAY IS 05-20-68 would be printed.

Example: Program run on May 20, 1968.

```
     INTEGER*4 NØW(2)
     CALL IDAY (NØW)
     PRINT 3, NØW  or  PRINT 3, NØW(1), NØW(2)
   3 FØRMAT (14H_THIS_DATE_IS_, 2A4)
```

THIS DATE IS 05-20-68 would be printed.

Function LØC

This function is used in the following manner:

I = LØC (X)

The result stored in the integer*4 location I is the core address of the argument X. This function can be useful in certain situations where storage of variables is not used in a straightforward manner.

Function MØDEL

This integer*4 function is used in the following manner:

M = MØDEL (0)

The function returns either the value 75 or the value 91 as the result depending on whether the computer the program is running on is the 360/75 or the 360/91. The argument of the function is not used, so 0 is suggested for use.

Example:

```
      M = MØDEL (0)
      PRINT 100, M
  100 FØRMAT('18H_CØMPUTER_IS_MØDEL, I3)
```

CØMPUTER IS MØDEL 75 would be printed out.

Masking Functions

It can be convenient to have a method for operating with one or more bits in a series of bits. For example, data from experiments usually have to be manipulated in order to test or compute with them. That is, experimental devices might write on tape 300 bits of information with each successive 15 bits containing one "piece of data". Manipulating or extracting these bits can be quite clumsy in FØRTRAN and many programmers prefer to write machine language subroutines to operate on the data to produce "System/360 meaningful" numbers. A halfway point between FØRTRAN coding and machine language coding is provided by the group of functions described here. They perform "logical" operations on words which are 32 bits in length (integer*4 or real*4 words). The four masking functions described may be either of type real*4 or type integer*4 depending on the name chosen. For example, there are the two fundtions AND and IAND. The result returned by either of the functions is identical. However, because of the FØRTRAN naming conventions, the FORTRAN compiler treats the function AND as a real*4 function and the function IAND as an integer*4 function. Thus: K=IAND(X,Y) and Z=AND(X,Y) would result in K and Z having identical bit patters; K=AND(X,Y) would cause the result returned by the AND function to be treated as a real*4 value and then converted to an integer*4 value before being stored in K. This is almost always undesirable; hence the use of two different names for functions performing the same logical operation. In all functions described any argument may be of type real*4 or integer*4. No conversion is performed on any argument.

Functions AND and IAND

A=AND(ARG1,ARG2) or I=IAND(ARG1,ARG2)

These functions return the logical product, bit by bit, of the two arguments. The product table is:

0x0=0

0x1=0

1x0=0

1x1=1

Functions ØR and IØR

A=ØR(ARG1,ARG2) or I=IØR(ARG1,ARG2)

These functions return the logical sum, bit by bit, of ARG1 and ARG2. The logical sum table is:

0+0=0

0+1=1

1+0=1

1+1=1

Functions XCØR and IXCØR

A=XCØR(ARG1,ARG2) or I=IXCØR(ARG1,ARG2)

These functions return the exclusive or (modulo-two sum), bit by bit, of ARG1 and ARG2. The exclusive or table is:

0+0=0

0+1=1

1+0=1

1+1=0

Functions CØMPL and ICØMPL

A=CØMPL(ARG1) or I=ICØMPL(ARG1)

The functions return the complement, bit by bit, of ARG1. The complement of 0 is 1 and the complement of 1 is 0.

Assume ARG1 and ARG2 have as the first four bits of their 32 bits those shown in the table below. The result of each function is given:

FUNCTIØN	AND-IAND	ØR-IØR	CØMPL-iCØMPL	XCØR-IXCØR
ARG1	1010	1010	1010	1010
ARG2	1100	1100	--	1100
RESULT	1000	1110	0101	0110

Character Manipulation Subroutines

In order to facilitate character manipulation using the Fortran language, a group of subroutines is automatically available for use from the system library.

A character string is defined as a consecutive group of bytes stored in memory. The last byte of any string is a byte having the value 00. This particular byte is called the terminator and is indicated by the character Δ in the following text. In particular, the characters of interest are normally those which can be keypunched. Tables of the byte representation of each character appear in several IBM/360 manual appendices. One convenient reference is the "green card", IBM System/360 Reference Data, Form X20-1703. The letter A has the hexadecimal representation C1, R is D9, 1 is F1, 9 is F9, $ is 5B. The hexadecimal representation 00 is reserved for use as the string terminator. The length of a string is the number of characters preceding the Δ. A string can have a length of 0 or any positive value depending solely on the computer storage set aside by the programmer. One method for defining storage for a string is use the type logical*1 for the string variable name. For example, the Fortran statement LØGICAL*1 X(25) would allow the string designated by X to have a maximum length of 24 bytes (one byte extra being used for the terminator).

Subroutine INSERT

This subroutine is used as follows:

CALL INSERT (STR, J, N, Q)

This might be "read" as: insert, beginning at the Jth character in the string STR, the first N characters from Q. If originally STR is AB12$Δ and Q is RSTΔ, J=4 and N=2; after the CALL statement, STR would consist of the string AB1RS2$Δ. Q does not necessarily have to be a string. The following is allowed:

CALL INSERT (STR, J, 2, 2HRS)

and has the same effect as in the first example.

STR may be a variable of any type declaration, for example, LØGICAL*1 STR(32) or REAL*8 STR(4). The only restriction is that the length of the string stored in STR may never be greater than 31 (for the declarations above). J is an integer*4 variable or a constant. If J is greater than the length of the string STR, then the characters from Q are inserted at the end of the string STR (immediately preceding the terminator). If STR is AB12$Δ and J is 6 or greater then with N=2 and Q being 2HRS, the resulting string in STR would be: AB12$RSΔ. If J is less than or equal to 0, a string of length N is formed: CALL INSERT (STR, 0, 2, 2HRS) would result in STR becoming RSΔ. If N is less than or equal to 0, no action is taken. If N is greater than the length of Q, unpredictable results will occur.

Subroutine INTØBC

This subroutine is used as follows:

CALL INTØBC (INT,STR)

The integer*4 value of INT is converted to EBCDIC and stored in string form at STR. The terminator Δ is placed after the string. A minus sign, for negative values of INT, will be the first character of STR. A plus sign is not used. This subroutine may also be called under the name INTØBCD: CALL INTØBCD (INT, STR).

INT	STR	HEX
1	1Δ	F100
-5	-5Δ	60F500
0	0Δ	F000
1237	1237Δ	F1F2F3F700

Subroutine INTBCD

This subroutine is used as follows:

CALL INTBCD (INT, STR, L)

The result of using this subroutine is the same as if the following two consecutive CALL statements were made:

CALL INTØBC (INT, STR)

CALL LENGTH (STR, L)

Thus, besides converting an integer to string form at STR, the length of the resulting string is also returned in the integer*4 variable L. This subroutine may also be called under the name INTBCDL, CALL INTBCDL (INT, STR, L).

Subroutine ERTRAN (I, D, T)

This routine generates a string of data containing the month, day, year and time on the UNIVAC-1108 when it is called. D contains the month, day and year, and T contains the time.

Function TICKER (DUM)

This function is used in the following manner: M=TICKER(0). The function returns the reading of the computer timer in sixtieth of seconds on the UNIVAC-1108.

Subroutine SECOND(T)

This routine is used as follows: CALL SECOND(T).
The result of this call is that the c.p.u. time on the CDC-6600 is returned in seconds in T.

4.9.3. Overlay Structure

To overlay MØRSE as described in this document the following arrangement may be used:

```
OVERLAY AA
INSERT INPUT1
INSERT SORIN.JOHIN.BNKNMC.GENI.GTVLIN.ALBERT
OVERLAY AA
INSERT INPUT2
OVERLAY BB
INSERT ALBIN.JNPUT.XSEC.ANGLES.BADMOM.FIND.GETMUS.LEGEND
INSERT MAMENT.Q.READSG.STORE.RESTOR.GTNDSK.GTSCT.XSTAPE.FFREAD
OVERLAY BB
INSERT SCORIN.STRUN.INSCOR.ENRGYS
OVERLAY AA
INSERT SOURCE.GPROB.GETETA.EUCLID.FLUXST.ALBDO.GSTORE.DIREC.PTHETA
INSERT TESTW.NBATCH.STBTCH.FBANK.FPROB.FSOUR.GOMST.MSOUR.NXTCOL
INSERT COLISN.GTIOUT.GEOM.LOOKZ.NORML.GOMFLP.G1.GG.PR
INSERT NRUN.VAR2.VAR3.ENDRUN
```

A diagram of the overlay structure is given in Fig. 4.21.

User routines called furing the random walk (e.g., SDATA, RELCØL, BDRYX) should be inserted in the ØVERLAY AA which begins with INSERT SØURCE, etc.

This overlay arrangement saves approximately 75K bytes out of 155K bytes.

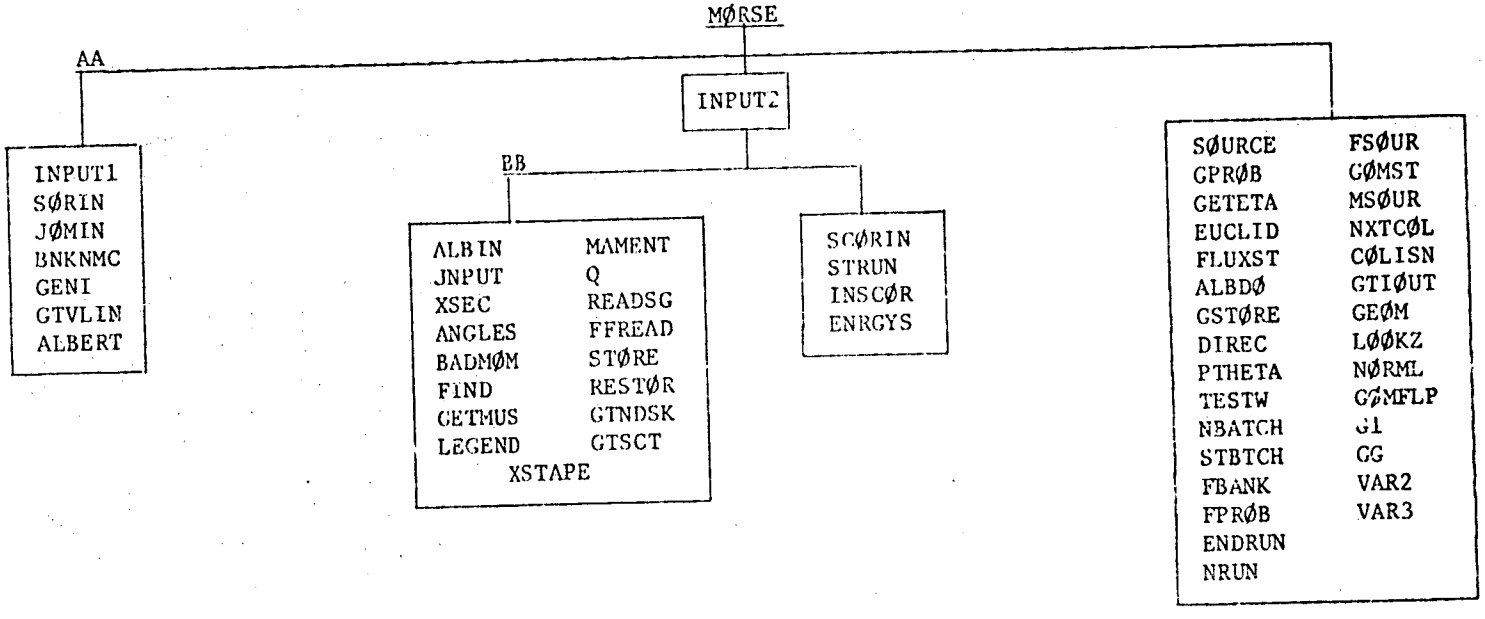


Fig. 4.21. MØRSE Overlay Structure

4.10 Many Integral Forms of the Boltzmann Transport Equation and its Adjoint

The purpose here is to derive a complete set of forward and adjoint integral transport equations in energy-group notation and to relate these equations to the Monte Carlo procedures used in the MØRSE code.

The Boltzmann Transport Equation

The derivation begins with the general time-dependent integro-differential form of the Boltzmann transport equation, the derivation of which can be regarded as a bookkeeping process that sets the net storage of particles within a differential element of phase space ($d\bar{r}dEd\bar{\Omega}$) equal to the particle gains minus particle losses in ($d\bar{r}dEd\bar{\Omega}$) and leads to the following familiar and useful form:

$$\frac{1}{v}\frac{\partial}{\partial t}\phi(\bar{r},E,\bar{\Omega},t) + \bar{\Omega}\cdot\nabla\phi(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)$$

$$= S(\bar{r},E,\bar{\Omega},t) + \int\int dE'd\bar{\Omega}'\,\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t) \qquad (1)$$

where

$(\bar{r},E,\bar{\Omega},t)$ denotes the general seven-dimensional phase space,

$\bar{r}$ = position variable,

E = the particle's kinetic energy,

v = the particle's speed corresponding to its kinetic energy E,

$\bar{\Omega}$ = a unit vector which describes the particle's direction of motion,

t = time variable,

$\phi(\bar{r},E,\bar{\Omega},t)$ = the time-dependent angular flux,

$\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = the number of particles that cross a unit area normal to the $\bar{\Omega}$ direction per unit time at the space point $\bar{r}$ and time t with energies in dE about E and with directions that lie within the differential solid angle $d\bar{\Omega}$ about the unit vector $\bar{\Omega}$,

$\frac{1}{v}\frac{\partial}{\partial t}\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = net storage (gains minus losses) per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and with directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$.

$\bar{\Omega}\cdot\nabla\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = net convective loss per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$\Sigma_t(\bar{r},E)$ = the total cross section at the space point $\bar{r}$ for particles of energy E,

$\Sigma_t(\bar{r},E)\phi(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = collision loss per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})dEd\bar{\Omega}$ = the differential scattering cross section which describes the probability per unit path that a particle with an initial energy E' and an initial direction $\bar{\Omega}'$ undergoes a scattering collision at $\bar{r}$ which places it into a direction that lies in $d\bar{\Omega}$ about $\bar{\Omega}$ with a new energy in dE about E,

$\left(\iint\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'\right)dEd\bar{\Omega}$ = inscattering gain per unit volume and time at the space point $\bar{r}$ and time t of particles with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$S(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = source particles emitted per unit volume and time at the space point $\bar{r}$ and time t with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$.

An effect of interest such as biological dose, energy deposition, or particle flux (denoted by λ) for a given problem can be expressed in terms of the flux field $\phi(\bar{r},E,\bar{\Omega},t)$ and an appropriate response function $P^{\phi}(\bar{r},E,\bar{\Omega},t)$ due to a unit angular flux and is given by:

$$\lambda = \iiiint P^{\phi}(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \ . \qquad (2)$$

Consistent with the MØRSE code, the energy dependence of Equation (1) will be represented in terms of energy groups which are defined such that:

ΔE_g = energy width of the gth group,

g = 1 corresponds to the highest energy group,

g = G corresponds to the lowest energy group,

with the obvious constraint that

$$\sum_{g=1}^{G}\Delta E_g = \int_{o}^{E_o} dE = E_o, \text{ the maximum particle energy.}$$

A "group" form of Equation (1) is obtained by integrating each term with respect to the energy variable over the energy interval ΔE_g:

$$\frac{\partial}{\partial t}\int_{\Delta E_g}\frac{1}{v}\,\phi(\bar{r},E,\bar{\Omega},t)dE + \bar{\Omega}\cdot\nabla\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE + \int_{\Delta E_g}\Sigma_t(\bar{r},E)\phi(\bar{r},E,\bar{\Omega},t)dE \tag{3}$$

$$= \int_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE + \sum_{g'=g}^{1}\int_{\Delta E_{g'}}\int_{4\pi} dE'd\bar{\Omega}'\int_{\Delta E_g}\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\phi(\bar{r},E',\bar{\Omega}',t)dE$$

Equation (3) provides the formal basis for the following group parameters:*

$\phi_g(\bar{r},\bar{\Omega},t)$ = time-dependent group angular flux,

$$= \int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE\ , \tag{4}$$

$\Sigma_t^g(\bar{r})$ = energy-averaged total cross section for the gth group,

$$\equiv \frac{\int_{\Delta E_g}\Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE} \tag{5}$$

v_g = energy-averaged particle speed for the gth group,

$$\equiv \frac{\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\frac{1}{v}\,\phi(\bar{r},E,\bar{\Omega},t)dE} \tag{6}$$

$\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})$ = group g' to group g scattering cross section,

$$\equiv \frac{\int_{\Delta E_{g'}}\int_{\Delta E_g}\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t)dE'dE}{\int_{\Delta E_{g'}}\phi(\bar{r},E',\bar{\Omega}',t)dE'} \tag{7}$$

$S_g(\bar{r},\bar{\Omega},t)$ = distribution of source particles for the gth group,

$$\equiv \int_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE\ . \tag{8}$$

The group form of the Boltzmann equation expressed in terms of the aforedefined group parameters is given by

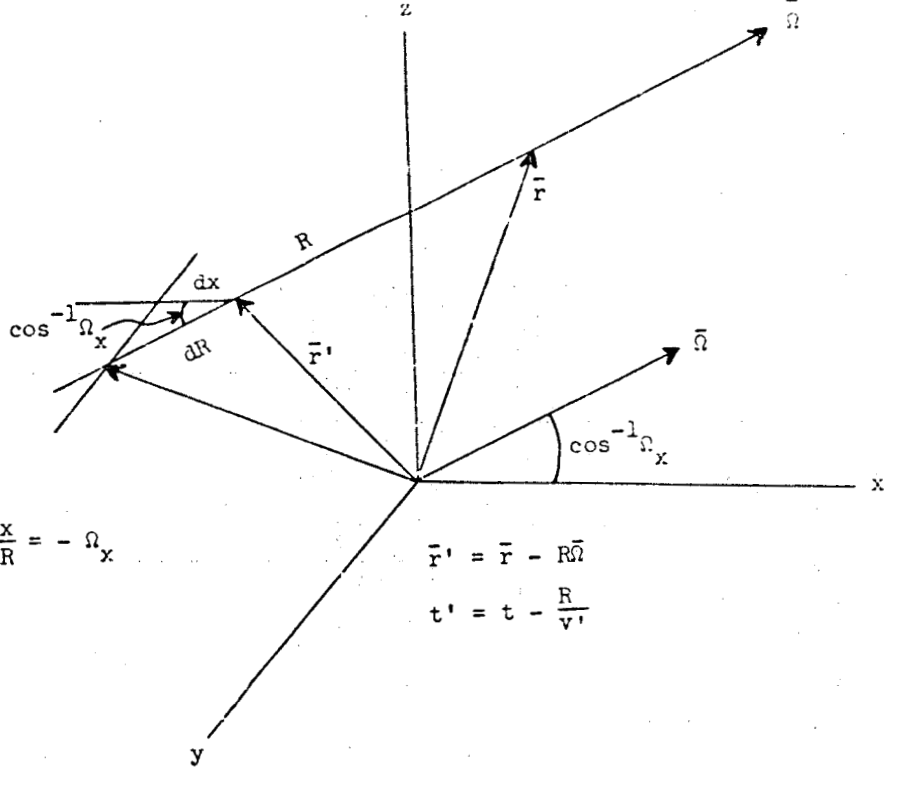
$$\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g(\bar{r},\bar{\Omega},t) + \bar{\Omega}\cdot\nabla\,\phi_g(\bar{r},\bar{\Omega},t) + \Sigma_t^g(\bar{r})\,\phi_g(\bar{r},\bar{\Omega},t)$$

$$= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\,\phi_{g'}(\bar{r},\bar{\Omega}',t)\,, \tag{9}$$

where the summation over energy groups could be expanded over all g' to allow for upscattering -- not usually considered important in shielding problems.

Integral Flux Density Equation

The transformation of Equation (9) into an integral form is now considered. To accomplish this, the combination of the convection and storage terms are first expressed in terms of the spatial variable R which relates a fixed point in space ($\bar{r}$) to an arbitrary point ($\bar{r}'$), as shown below.



The total derivative of the angular flux with respect to R is given by

$$\frac{d}{dR}\,\phi(\bar{r}',E,\bar{\Omega},t') = \frac{\partial x}{\partial R}\frac{\partial \phi}{\partial x} + \frac{\partial y}{\partial R}\frac{\partial \phi}{\partial y} + \frac{\partial z}{\partial R}\frac{\partial \phi}{\partial z} + \frac{\partial t}{\partial R}\frac{\partial \phi}{\partial t}$$

which, according to Fig. A.1 and noting that the particle's speed (v) is equal to (- dR/dt) can be rewritten as

$$\begin{aligned}\frac{d}{dR}\,\phi(\bar{r}',E,\bar{\Omega},t') &= -\,\Omega_x\frac{\partial \phi}{\partial x} - \Omega_y\frac{\partial \phi}{\partial y} - \Omega_z\frac{\partial \phi}{\partial z} - \frac{1}{v}\frac{\partial \phi}{\partial t} \\ &= -\,\bar{\Omega}\cdot\nabla\,\phi(\bar{r}',E,\bar{\Omega},t') - \frac{1}{v}\frac{\partial \phi}{\partial t}\,.\end{aligned} \tag{10}$$

Equation (10) can be expressed in group notation as

$$-\frac{d}{dR}\,\phi_g(\bar{r}',\bar{\Omega},t') = \frac{1}{v_g}\frac{\partial}{\partial t}\,\phi_g(\bar{r}',\bar{\Omega},t') + \bar{\Omega}\cdot\nabla\,\phi_g(\bar{r}',\bar{\Omega},t')\,. \tag{11}$$

Substitution of Eq. (11) into Eq. (9) with $\bar{r} \equiv \bar{r}'$ and $t \equiv t'$ yields

$$\begin{aligned}-\frac{d}{dR}\,\phi_g(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\,\phi_g(\bar{r}',\bar{\Omega},t') &= S_g(\bar{r}',\bar{\Omega},t) \\ &+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\,\phi_{g'}(\bar{r}',\bar{\Omega},t)\,.\end{aligned} \tag{12}$$

The integrating factor

$$e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

is introduced in the following manner:

$$\begin{aligned}\frac{d}{dR}\left\{\phi_g(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right\} &= -\,e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} \\ &\times\left\{-\frac{d\phi_g}{dR} + \Sigma_t^g(\bar{r}')\,\phi_g(\bar{r}',\bar{\Omega},t')\right\}\,.\end{aligned} \tag{13}$$

Using Eq. (13), Eq. (12) can be rewritten as

$$-\frac{d}{dR}\left[\phi_g(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'}\right] = e^{-\int_0^R \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'} \times \left[S_g(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega},t')\right]. \tag{14}$$

Multiply Eq. (14) by dR and integrate (R = 0 to R = ∞); then

$$\phi_g(\bar{r},\bar{\Omega},t) - \phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'} = \int_0^\infty dR\, e^{-\int_0^R \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'}\left[S_g(\bar{r}-R\bar{\Omega},\bar{\Omega},t-\tfrac{R}{v}) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r}-R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right] \tag{15}$$

Require that

$$\left[\phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'}\right] = 0\,, \tag{16}$$

and introduce the "optical thickness"

$$\beta_g(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r}-R'\bar{\Omega})dR'\,, \tag{17}$$

and Eq. (15) becomes

$$\phi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g(\bar{r},R,\bar{\Omega})}\left[S_g(\bar{r}-R\bar{\Omega},\bar{\Omega},t-R/v) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r}-R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right]. \tag{18}$$

Equation (18) will be referred to as the "Integral Flux Density Equation."

An effect of interest λ in group notation can be expressed as

$$\lambda_g = \int\int\int\int P_g^{\phi}(\bar{r},\bar{\Omega},t)\ \phi_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \quad , \tag{19}$$

where

$P_g^{\phi}(\bar{r},\bar{\Omega},t)$ = the response function of the effect of interest due to a unit angular group flux (group g, $\bar{r}$, $\bar{\Omega}$, time t),

$$= \frac{\int_{\Delta E_g} P^{\phi}(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE}$$

λ_g = that portion of the effect of interest associated with the gth energy group.

The λ_g are so defined that the total effect of interest λ is given by the summation

$$\lambda = \sum_{g=1}^{G} \lambda_g \ . \tag{20}$$

Integral Event Density Equation

The "event density" $\psi_g(\bar{r},\bar{\Omega},t)$ describes the density of particles going into a collision and is related to the group angular flux in the following manner:

$$\psi_g(\bar{r},\bar{\Omega},t) \equiv \Sigma_t^g(\bar{r})\ \phi_g(\bar{r},\bar{\Omega},t) \ . \tag{21}$$

where

$\psi_g(\bar{r},\bar{\Omega},t)d\bar{\Omega}$ = the number of collision events per unit volume and time at the space point $\bar{r}$ and time t experienced by particles having energies within the gth energy group and directions in $d\bar{\Omega}$ about $\bar{\Omega}$.

The defining equation for the event density is obtained by multiplying both sides of Eq. (18) by the group total cross section $\Sigma_t^g(\bar{r})$ and identifying the product $\Sigma_t^g(\bar{r})\phi_g(\bar{r},\bar{\Omega},t)$ as the event density $\psi_g(\bar{r},\bar{\Omega},t)$:

$$\psi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, \Sigma_t^g(\bar{r})\, e^{-\tau_g(\bar{r},R,\bar{\Omega})} \left[S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - R/v) \right.$$

$$\left. + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \frac{\Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r}')}\, \psi_{g'}(\bar{r}',\bar{\Omega}',t') \right]. \tag{22}$$

Equation (22) will be referred to as the "Integral Event Density Equation."

The effect of interest λ_g can be expressed in terms of the event density; consider Eq. (19) rewritten as

$$\lambda_g = \iiint \frac{P_g^{\phi}(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})}\, \Sigma_t^g(\bar{r})\, \phi(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt$$

$$= \iiint P_g^{\psi}(\bar{r},\bar{\Omega},t)\, \psi_g(\bar{r},\bar{\Omega},t)\, d\bar{r}\, d\bar{\Omega}\, dt\ , \tag{23}$$

where

$P_g^{\psi}(\bar{r},\bar{\Omega},t)$ = the response function of the effect of interest due to a particle which experiences an event at (group g, $\bar{r}$, $\bar{\Omega}$, time t),

$P_g^{\psi}(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t)/\Sigma_t^g(\bar{r})$

or

$$P_g^{\phi}(\bar{r},\bar{\Omega},t) = \Sigma_t^g(\bar{r})\, P_g^{\psi}(\bar{r},\bar{\Omega},t)\ . \tag{24}$$

Integral Emergent Particle Density Equation

Define the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ as the density of particles leaving a source or emerging from a real collision with phase space coordinates (group g, $\bar{r}$, $\bar{\Omega}$, t),

$$\chi_g(\bar{r},\bar{\Omega},t) = S_g(\bar{r},\bar{\Omega},t) + \sum_{g'} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\, \psi_{g'}(\bar{r},\bar{\Omega}',t)\ . \tag{25}$$

Then Eq. (18) can be written as

$$\phi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g(\bar{r},R,\bar{\Omega})} \chi_g(\bar{r}',\bar{\Omega},t') \ . \tag{26}$$

The "Integral Emergent Particle Density Equation" is obtained by substituting Eq. (26) into Eq. (25):

$$\begin{aligned}
&\chi_g(\bar{r},\bar{\Omega},t) \\
&= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) \int_0^\infty dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t') \\
&= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \int_0^\infty dR\, \Sigma_t^{g'}(\bar{r})\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t').
\end{aligned} \tag{27}$$

The effect of interest λ_g can also be expressed in terms of the emergent particle density

$$\lambda_g = \iiint p_g^{\chi}(\bar{r},\bar{\Omega},t)\, \chi_g(\bar{r},\bar{\Omega},t) d\bar{r} d\bar{\Omega} dt \ . \tag{28}$$

The response function $p_g^{\chi}(\bar{r},\bar{\Omega},t)$ is obtained by considering a particle which emerges from a collision at $\bar{r}$ with phase space coordinates (group g, $\bar{\Omega}$, time t). This particle will experience an event in dR about $\bar{r}' = \bar{r} + R\bar{\Omega}$ at time $t' = t + R/v$ with the probability

$$\Sigma_t^g(\bar{r}')\, e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'} dR \ ,$$

and the contribution of this event is the response function $p_g^{\psi}(\bar{r}',\bar{\Omega},t')$. The sum of all such contributions to the effect of interest is given by

$$\int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'} p_g^{\psi}(\bar{r}',\bar{\Omega},t') \ ,$$

and should be the same as a response function $p_g^{\chi}(\bar{r},\bar{\Omega},t)$ which is based on emergent particle density. This leads to the following relationship:

$$p_g^{\chi}(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta^*(\bar{r},R,\bar{\Omega})} p_g^{\psi}(\bar{r}',\bar{\Omega},t') \ , \tag{29}$$

where

$P_g^{\chi}(\bar{r},\bar{\Omega},t)$ ≡ the response function (of the effect of interest due to a particle which emerges from a collision having the phase space coordinates (group g, $\bar{r}$, $\bar{\Omega}$, time t)

$$\beta^*(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR' \ . \tag{30}$$

It is noted that $\beta_g^*(\bar{r},R,\bar{\Omega})$ differs from the optical thickness $\beta_g(\bar{r},R,\bar{\Omega})$ as defined by Eq. (17) in that the integration is performed in the positive $\bar{\Omega}$ direction and as such $\beta_g^*(\bar{r},R,\bar{\Omega})$is the adjoint of $\beta_g(\bar{r},R,\bar{\Omega})$. $P_g^{\chi}(\bar{r},\bar{\Omega},t)$ can also be expressed in terms of $P_g^{\phi}(\bar{r},\bar{\Omega},t)$ by substituting Eq. (24) into Eq. (29), yielding

$$P_g^{\chi}(\bar{r},\bar{\Omega},t) = \int_0^{\infty} dR\ e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} P_g^{\phi}(\bar{r}',\bar{\Omega},t') \ . \tag{31}$$

Operator Notation and Summary of the Forward Equations

Define the transport integral operator

$$T_g(\bar{r}'\to\bar{r},\bar{\Omega}) \equiv \int_0^{\infty} dR\ \Sigma_t^g(\bar{r})\ e^{-\beta_g(\bar{r},R,\bar{\Omega})} \ , \tag{32}$$

and the collision integral operator

$$C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) = \sum_{g'=g}^{1} \int d\bar{\Omega}' \ \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \ , \tag{33}$$

which can be rewritten as

$$C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) = \sum_{g'=g}^{1} \int d\bar{\Omega}' \left[\frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_s^{g'}(\bar{r})}\right]\left[\frac{\Sigma_s^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\right], \tag{34}$$

where

$$\Sigma_s^{g'}(\bar{r}) = \sum_g \int d\bar{\Omega}\ \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) \ . \tag{35}$$

In Eq. (34), $[\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})/\Sigma_s^{g'}(\bar{r})]$ is a normalized probability density function from which the selection of a new energy group and direction can be accomplished and $[\Sigma_s^{g'}(\bar{r})/\Sigma_t^{g'}(\bar{r})]$ is the nonabsorption probability.

Using the transport and collision integral operators, Eq. (22) can be rewritten as

$$\psi_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}'\to\bar{r},\bar{\Omega})\left(S_g(\bar{r}',\bar{\Omega},t') + C_{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\psi_g(\bar{r}',\bar{\Omega}',t')\right). \quad (36)$$

The term $T_g(\bar{r}',\bar{r},\bar{\Omega})S_g(\bar{r}',\bar{\Omega},t')$ can be identified as the "first collision source" and denoted by

$$S_c^g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}'\to\bar{r},\bar{\Omega})S_g(\bar{r}',\bar{\Omega},t') , \quad (37)$$

and the "Integral Event Density Equation" becomes

$$\psi_g(\bar{r},\bar{\Omega},t) = S_c^g(\bar{r},\bar{\Omega},t) + T(\bar{r}'\to\bar{r},\bar{\Omega})C_{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\psi_g(\bar{r}',\bar{\Omega}',t') . \quad (38)$$

Using the relationship $\psi_g(\bar{r},\bar{\Omega},t) = \Sigma_t^g(\bar{r})\phi_g(\bar{r},\bar{\Omega},t)$, Eq. (38) can be transformed into the "Integral Flux Density Equation:"

$$\phi_g(\bar{r},\bar{\Omega},t) = \frac{S_c^g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} + T_g(\bar{r}'\to\bar{r},\bar{\Omega})C_{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\frac{\Sigma_t^{g'}(\bar{r}')}{\Sigma_t^g(\bar{r})}\phi_{g'}(\bar{r}',\bar{\Omega}',t'). \quad (39)$$

Finally, the integral operators are introduced into Eq. (27) and the following form for the "Integral Emergent Particle Density Equation" is obtained:

$$\chi_g(\bar{r},\bar{\Omega},t) = S_g(\bar{r},\bar{\Omega},t) + C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\ T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}')\ \chi_{g'}(\bar{r}',\bar{\Omega}',t'). \quad (40)$$

An examination of Equations (38), (39), and (40) would reveal that either the "Integral Event Density Equation" or the "Integral Emergent Particle Density Equation" would provide a reasonable basis for a Monte Carlo random walk. Equation (40) was selected for the MØRSE code since the source particles would be introduced according to the natural distribution rather than the distribution of first collisions. However, it is noted that after the introduction of the source particle, the subsequent

random walk can be regarded in terms of either Eq. (38) or Eq. (40) with the particle's weight at a collision site being the weight before collision (WTBC) or the weight after collision (WATE), respectively.

The random walk based on the "Integral Emergent Particle Density Equation" would introduce a particle into the system according to the source function. The particle travels to the site of its first collision as determined by the transport kernel. Its weight is modified by the non-absorption probability and a new energy group and flight direction are selected from the collision kernel. The transport and collision kernels are applied successively determining the particle's emergent phase space coordinates corresponding to the second, third, etc., collision sites until the random walk is terminated due to the reduction of the particle's weight below some cut-off value or because the particle escapes from that portion of phase space associated with a particular problem (for example, escape from the system, slowing down below an energy cut-off, or exceeding some arbitrarily specified age cut-off).

Random Walk Procedure

The actual implementation of the random walk procedure is accomplished by approximating the integrals implied in the collision and transport integral operators by the sum

$$\chi_g(\bar{r},\bar{\Omega},t) = \sum_{n=0}^{\infty} \chi_g^n(\bar{r},\bar{\Omega},t) \quad , \tag{41}$$

where

$\chi_g^n(\bar{r},\bar{\Omega},t)d\bar{\Omega}$ = the emergent particle density of particles emerging from its nth collision and having phase space coordinates (group g, $\bar{r}$, $d\bar{\Omega}$ about $\bar{\Omega}$, time t),

$\chi_g^o(\bar{r},\bar{\Omega},t) = S_g(\bar{r},\bar{\Omega},t)$,

$\chi_g^n(\bar{r},\bar{\Omega},t) = C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\ T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}')\chi_g^{n-1}(\bar{r}',\bar{\Omega}',t')$.

Thus, the source coordinates (group g_o, $\bar{r}_o$, $\bar{\Omega}_o$, time t_o) are selected from $S_g(\bar{r},\bar{\Omega},t)$ and a flight distance R is picked $\Sigma_t^{g_o}(\bar{r})e^{-\tau_{g_o}(\bar{r},R,\bar{\Omega}_o)}$ to determine the site for the first collision $\bar{r}_1$ and the particle's age $t_1 = t_o + R/v_{g_o}$. The probability of scattering is $\Sigma_s^{g_o}(\bar{r}_1)/\Sigma_t^{g_o}(\bar{r}_1)$. All particles are forced to scatter and their weight is modified with this probability. A new group g_1 is selected according to the distribution

$$\frac{\int_{4\pi} d\bar{\Omega}\ \Sigma_s^{g_o \to g}(\bar{r}, \bar{\Omega}_o \to \bar{\Omega})}{\Sigma_s^{g_o}(\bar{r}_1)} .$$

and then a new direction $\bar{\Omega}$ is determined from

$$\frac{\Sigma_s^{g_o \to g_1}(\bar{r}_1, \bar{\Omega}_o \to \bar{\Omega})}{\Sigma_s^{g_o \to g_1}(\bar{r}_1)} .$$

The process is repeated until the particle history is terminated. Contributions to the quantity of interest are estimated at appropriate points in the random walk (boundary crossings, before or after real collisions, etc.) using the particle's WATE and the estimator $P_g^X(\bar{r},\bar{\Omega},t)$.

Derivation of the Adjoint Integro-Differential Boltzmann Transport Equation

Consider a (as yet unspecified) function $\phi^*(\bar{r},E,\bar{\Omega},t)$ which exists over the same phase space and satisfies the same kind of boundary conditions satisfied by the forward angular flux $\phi(\bar{r},E,\bar{\Omega},t)$. Further, let an operator O^* be defined such that the following integral relationship is satisfied:

$$\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ O\ \phi(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt$$

$$= \iiiint \phi(\bar{r},E,\bar{\Omega},t)\ O^*\ \phi^*(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt + \text{(Boundary Terms)} .$$

The O^* operator will be referred to as the adjoint operator to the corresponding forward operator O.

Multiply each term of the Boltzmann transport equation, Eq. (1), by the function $\phi^*(\bar{r},E,\bar{\Omega},t)$ and integrate the resultant equation (term by term) over all phase space:

$$\begin{aligned} &\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \frac{1}{v}\frac{\partial}{\partial t}\ \phi(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \\ &\quad \times \nabla\cdot\bar{\Omega}\ \phi(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E) \\ &\quad \times \phi(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt = \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ S(\bar{r},E,\bar{\Omega},t) d\bar{r}dEd\bar{\Omega}dt \\ &\quad + \iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t) dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt . \end{aligned} \tag{42}$$

It can be shown that the following adjoint relationships are true for the conditions associated with a particle transport problem:

$$\iiiii \phi^*(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\,\frac{\partial}{\partial t}\,\phi(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt = -\iiiii \phi(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\,\frac{\partial}{\partial t} \times \phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt + \left\{\iiiii \frac{1}{v}\,\frac{\partial}{\partial t}\,\phi\,\phi^*\,d\bar{r}dEd\bar{\Omega}dt\right\}_{\text{Boundary Term}} \tag{43}$$

$$\iiiii \phi^*(\bar{r},E,\bar{\Omega},t)\,\Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega}t)\,d\bar{r}dEd\bar{\Omega}dt = \iiiii \phi(\bar{r},E,\bar{\Omega},t) \times \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt\ , \tag{44}$$

$$\iiiii \phi^*(\bar{r},E,\bar{\Omega},t)\,\nabla\cdot\bar{\Omega}\,\phi(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt = -\iiiii \phi(\bar{r},E,\bar{\Omega},t) \times \nabla\cdot\bar{\Omega}\,\phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt + \left\{\iiiii \phi\,\phi^*\bar{\Omega}\cdot d\bar{r}dEd\bar{\Omega}dt\right\}_{\text{Boundary Term}} \tag{45}$$

$$\iiiii \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt = \iiiii \phi(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt \tag{46}$$

The boundary terms which occur in Equations (43) and (45) may be made to vanish while conforming to the natural characteristics of the system under analysis. For example, the extent of the time domain can be defined such that initial and final values of ϕ and/or ϕ^* are zero [and the boundary term of Eq. (43) vanishes]. Also, the surface within which the spatial domain of phase space is contained can be so located that the combination $[\phi\ \phi^*]$ is zero everywhere on that surface [and the boundary term of Eq. (45) vanishes]. For most Monte Carlo analyses, the elimination of the boundary terms in no way restricts the generality of the solution obtained.

Using the adjoint relationships given by Equations (43) through (46), and presuming that the boundary terms vanish, Eq. (42) can be rewritten as

$$-\iiiii \phi(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\,\frac{\partial}{\partial t}\,\phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt - \iiiii \phi(\bar{r},E,\bar{\Omega},t) \times \nabla\cdot\bar{\Omega}\,\phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt + \iiiii \phi(\bar{r},E,\bar{\Omega},t)\,\Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt = \iiiii \phi(\bar{r},E,\bar{\Omega},t)\,S^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt + \iiiii \phi(\bar{r},E,\bar{\Omega},t) \times \iint \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt\ , \tag{47}$$

where the adjoint source term $S^*(\bar{r},E,\bar{\Omega},t)$ is defined such that

$$\int\int\int\int\int \phi(\bar{r},E,\bar{\Omega},t)\, S^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt = \int\int\int\int\int \phi^*(\bar{r},E,\bar{\Omega},t)\, S(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt \ . \tag{48}$$

Noting that the forward flux $\phi(\bar{r},E,\bar{\Omega},t)$ can be factored from each term, Eq. (47) can be rearranged as follows:

$$\int\int\int\int\int \phi(\bar{r},E,\bar{\Omega},t) \left\{ -\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\,\phi^*(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t) - S^*(\bar{r},E,\bar{\Omega},t) - \int\int \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}') \times \phi^*(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}' \right\} d\bar{r}dEd\bar{\Omega}dt = 0 \ . \tag{49}$$

It is required that the forward angular flux $\phi(\bar{r},E,\bar{\Omega},t)$ correspond to non-trivial physical situations, i.e., $\phi(\bar{r},E,\bar{\Omega},t) > 0$ over at least some portion of phase space. The observation is made that $\phi^*(\bar{r},E,\bar{\Omega},t)$ is still essentially undefined and that many functions $\phi^*(\bar{r},E,\bar{\Omega},t)$ probably satisfy Eq. (49). At this point, $\phi^*(\bar{r},E,\bar{\Omega},t)$ is defined to be that function which satisfies the following equation:

$$\left\{ -\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\phi^*(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\phi^*(\bar{r},E,\bar{\Omega},t) - S^*(\bar{r},E,\bar{\Omega},t) - \int\int \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}' \right\} = 0 \ .$$

This condition also satisfies Eq. (49) exactly and provides the following $\phi^*(\bar{r},E,\bar{\Omega},t)$-defining integro-differential equation:

$$-\frac{1}{v}\frac{\partial}{\partial t}\phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\,\phi^*(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t) = S^*(\bar{r},E,\bar{\Omega},t) + \int\int \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)\,dE'd\bar{\Omega}' \ , \tag{50}$$

which is commonly called the "Adjoint Integro-Differential Boltzmann Equation." However, it will not be the practice here to refer to the function $\phi^*(\bar{r},E,\bar{\Omega},t)$

as the adjoint flux; consistent terminology will be introduced later in this section.

At this point, two procedures for defining and calculating group adjoint fluxes are considered. One method involves integrating each term of Eq. (50) over the energy interval ΔE_g, which leads to the following group equations:

$$-\frac{1}{\hat{v}_g}\frac{\partial}{\partial t}\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\,\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) + \hat{\Sigma}^g_t(\bar{r})\,\hat{\phi}^*_g(\bar{r},\bar{\Omega},t)$$
$$= S^*_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{G}\int d\bar{\Omega}'\,\hat{\Sigma}^{g\to g'}_s(\bar{r},\bar{\Omega}\to\bar{\Omega}')\,\hat{\phi}^*_{g'}(\bar{r},\bar{\Omega}',t)\,, \qquad g = 1,2,\ldots G \tag{51}$$

where

$$\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) = \frac{1}{\Delta E_g}\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE\,, \tag{52}$$

$$\hat{v}_g = \frac{\displaystyle\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\displaystyle\int_{\Delta E_g}\frac{1}{v}\,\phi^*(\bar{r},E,\bar{\Omega},t)dE} \tag{53}$$

$$\hat{\Sigma}^g_t(\bar{r}) \equiv \frac{\displaystyle\int_{\Delta E_g}\Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\displaystyle\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}\,, \tag{54}$$

$$\hat{\Sigma}^{g\to g'}_s(\bar{r},\bar{\Omega}\to\bar{\Omega}') \equiv \frac{\displaystyle\int_{\Delta E_g}\int_{\Delta E_{g'}}\Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)dE'dE}{\displaystyle\int_{\Delta E_{g'}}\phi^*(\bar{r},E',\bar{\Omega}',t)dE'}\,, \tag{55}$$

$$\hat{S}^*_g(\bar{r},\bar{\Omega},t) = \frac{1}{\Delta E_g}\int_{\Delta E_g}S^*(\bar{r},E,\bar{\Omega},t)dE\,. \tag{56}$$

The $\hat{\Sigma}_t^g(\bar{r})$, $\hat{\Sigma}_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')$, and $\hat{v}_g$ are adjoint weighted group parameters and their use in the solution of Eq. (51) provides group adjoint fluxes defined by Eq. (52) where $\phi^*(\bar{r},E,\bar{\Omega},t)$ represents the solution of Eq. (50).

Another approach for defining group adjoint fluxes is to directly devise the equation which is adjoint to the group form of the Boltzmann equation [Eq. (9)]. The group adjoint equation so obtained* is given by

$$-\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g^*(\bar{r},\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\phi_g^*(\bar{r},\bar{\Omega},t) + \Sigma_t^g(\bar{r})\phi_g^*(\bar{r},\bar{\Omega},t)$$
$$= S_g^*(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{G}\int d\bar{\Omega}'\ \Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\phi_g^*(\bar{r},\bar{\Omega}',t)\ , \qquad (57)$$
$$g = 1,2,\ldots G.$$

where v_g, $\Sigma_t^g(\bar{r})$ are forward weighted group parameters identified to those which occur in Eq. (9) and the matrix $\Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')$ is simply the transposition of the forward weighted group-to-group differential scattering cross-section matrix.

The group adjoint fluxes $\phi_g^*(\bar{r},\bar{\Omega},t)$ which represent the solution of Eq. (57) are adjoint to the group fluxes ϕ_g and do not necessarily assume the same values as the group adjoint fluxes $\hat{\phi}_g^*(\bar{r},\bar{\Omega},t)$, i.e.,

$$\phi_g^*(\bar{r},\bar{\Omega},t) \neq \frac{1}{\Delta E_g}\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE\ .$$

This follows since $\Sigma_t^g(\bar{r})$, $\Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')$, and v_g are, in general, different from the adjoint weighted values. Usually forward weighted group parameters, as implied by Eq. (57), are used in MORSE. However, other weighting schemes, such as adjoint or adjoint and forward, deserve consideration when cross-section weighting is a problem. When a sufficiently fine group structure is employed, the group parameters become less sensitive to the weighting scheme and the corresponding group adjoint fluxes are also nearly the same.

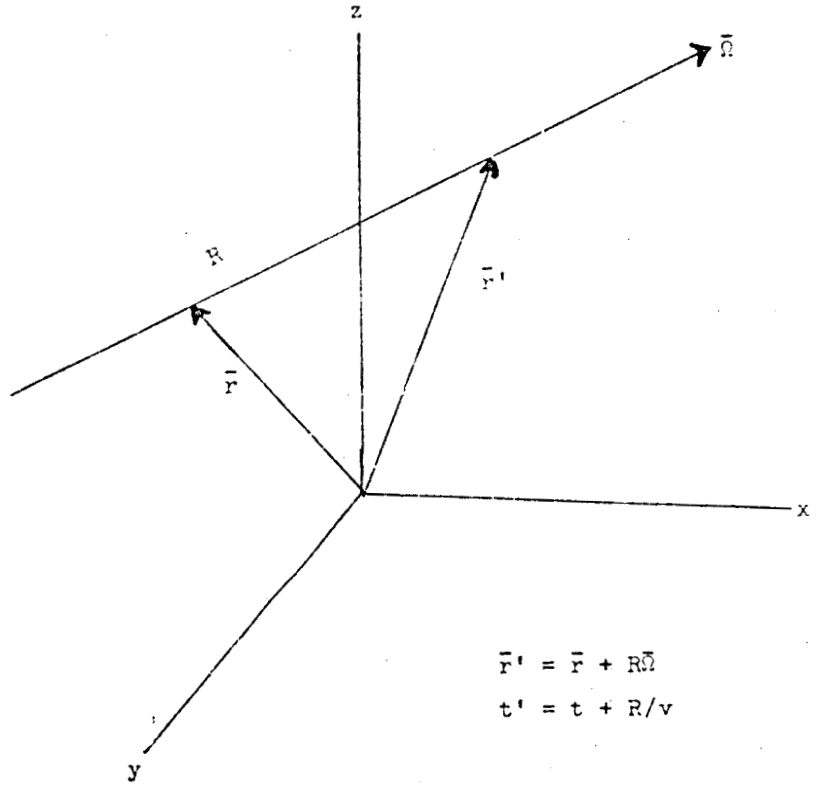
---

*The derivation of Eq. (57) is not presented here because of its similarity with the previous derivation of Eq. (50); the integrals over energy are simply replaced by appropriate group summations.

Integral Point-Value Equation

Equation (57) is now transformed into an integral form following essentially the same procedures used with the forward equations. As shown below, let $\bar{r}' = \bar{r} + R\bar{\Omega}$ rather than $\bar{r}' = \bar{r} - R\bar{\Omega}$ as was the convention with the forward equations. The total derivative of $\phi_g^*(\bar{r}',\bar{\Omega},t)$ with respect to R is given by

$$\frac{d}{dR}\,\phi_g^*(r',\bar{\Omega},t') = \frac{1}{v_g}\,\frac{\partial}{\partial t}\,\phi_g^*(r',\bar{\Omega},t') + \nabla\cdot\bar{\Omega}\;\phi_g^*(r',\bar{\Omega},t') \quad . \tag{58}$$



$$\bar{r}' = \bar{r} + R\bar{\Omega}$$

$$t' = t + R/v$$

$$- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'$$

Use of the integrating factor $e^{\qquad\qquad}$ provides the following relationship:

$$\frac{d}{dR}\left\{\phi_g^*(\bar{r}',\bar{\Omega},t')\, e^{- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\right\}$$

$$= \frac{d\phi_g^*}{dR}(\bar{r}',\bar{\Omega},t')\, e^{- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} - \Sigma_t^g(\bar{r}')\, \phi_g^*(\bar{r}',\bar{\Omega},t') \tag{59}$$

$$\times e^{- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} = e^{- \int_0^R \Sigma_t^g(\bar{r}' + R'\bar{\Omega})dR'}$$

$$\times \left[\frac{d}{dR}\phi_g^*(\bar{r}',\bar{\Omega},t') - \Sigma_t^g(\bar{r}')\, \phi_g^*(\bar{r}',\bar{\Omega},t')\right] .$$

Equation (59), together with Eq. (58), can be arranged to give

$$\left(-\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g^*(\bar{r}',\bar{\Omega},t') - \nabla\cdot\bar{\Omega}\,\phi_g^*(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')\right)$$

$$+ \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR' \qquad - \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR' \tag{60}$$

$$= e^{\qquad\qquad} \frac{d}{dR}\left[\phi_g^*(\bar{r}',\bar{\Omega},t')\, e^{\qquad\qquad}\right].$$

It is noted that Eq. (60) is identically the left-hand side of Eq. (57) which can now be rewritten as

$$- \frac{d}{dR}\left[\phi_g^*(\bar{r}',\bar{\Omega},t')e^{- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\right] = e^{- \int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} \tag{61}$$

$$\times \{S_g^*(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{G} \int d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}^*(\bar{r}',\bar{\Omega}',t')\}.$$

Integrate Eq. (61) from R = 0 to R = ∞ and assume that

$$\{\phi_g^*(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR'}\} \equiv 0 \; ; \tag{62}$$

then the following integral expression for $\phi_g^*(\bar{r},\bar{\Omega},t)$ is obtained:

$$\phi_g^*(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\{S_g^*(\bar{r}+R\bar{\Omega},\bar{\Omega},t+R/v_g) + \sum_{g'}\int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r}+R\bar{\Omega},\bar{\Omega}\to\bar{\Omega}')\,\phi_{g'}^*(\bar{r}',\bar{\Omega}',t')\} \; . \tag{63}$$

Equation (63) contains the adjoint optical thickness $\beta_g^*(\bar{r},R,\bar{\Omega})$ which was defined earlier by Eq. (30) as

$$\beta_g^*(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r}+R'\bar{\Omega})dR' \; .$$

Redefine the source term as

$$S_{Tg}^*(\bar{r},\bar{\Omega},t) = \int_0^R dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} S_g^*(\bar{r}+R\bar{\Omega},\bar{\Omega},t+R/v_g) \; , \tag{64}$$

and Eq. (63) can be rewritten as

$$\phi_g^*(\bar{r},\bar{\Omega},t) = S_{Tg}^*(\bar{r},\bar{\Omega},t) + \int dR\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} \sum_{g'}\int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \times \phi_{g'}^*(\bar{r}',\bar{\Omega}',t') = S_{Tg}^*(\bar{r},\bar{\Omega},t) + \int_0^\infty dR\, \Sigma_t^g(\bar{r}+R\bar{\Omega})\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} \times \sum_{g'}\int d\bar{\Omega}'\, \frac{\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^g(\bar{r}')}\, \phi_{g'}^*(\bar{r}',\bar{\Omega}',t') \; , \tag{65}$$

and in terms of the transport and collision operators, Eq. (65) becomes

$$\phi^*_g(\bar{r},\bar{\Omega},t) = S^*_{Tg}(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\, \phi^*_{g'}(\bar{r}',\bar{\Omega}',t') \quad . \quad (66)$$

A comparison of Eq. (66) with Equations (38), (39), and (40) reveals that the function $\phi^*_g(\bar{r},\bar{\Omega},t)$ as defined by Eq. (66) is adjoint to the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ as defined by Eq. (40). Therefore, let $\phi^*_g(\bar{r},\bar{\Omega},t)$ be denoted by $\chi^*_g(\bar{r},\bar{\Omega},t)$ and Eq. (66) becomes

$$\chi^*_g(\bar{r},\bar{\Omega},t) = S^*_{Tg}(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\, \chi^*_{g'}(\bar{r}',\bar{\Omega}',t') \quad . \quad (67)$$

The nature of $\chi^*_g(\bar{r},\bar{\Omega},t)$ will depend on $S^*_{Tg}(\bar{r},\bar{\Omega},t)$ -- how or on what basis should $S^*_{Tg}(\bar{r},\bar{\Omega},t)$ be specified? If $S^*(\bar{r},E,\bar{\Omega},t)$ is set equal to $P^{\phi}(\bar{r},E,\bar{\Omega},t)$ (the response function of the effect of interest λ due to a unit angular flux), then

$$\iiiint \phi(\bar{r},E,\bar{\Omega},t)\, S^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = \iiiint \phi(\bar{r},E,\bar{\Omega},t) \times P^{\phi}(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = \lambda \quad . \quad (68)$$

According to Eq. (48), the effect of interest λ would also be given by

$$\lambda = \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\, S(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \quad . \quad (69)$$

The effect of interest as given by Eq. (69) can also be expressed in group notation

$$\lambda = \sum_g \iiint \hat{\phi}^*_g(\bar{r},\bar{\Omega},t)\hat{S}_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt$$

$$= \sum_g \iiint \phi_g(\bar{r},\bar{\Omega},t)\hat{S}^*_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \quad ,$$

where

$\hat{\phi}^*_g(\bar{r},\bar{\Omega},t)$ is the group adjoint flux corresponding to the adjoint weighted group parameters,

$$\hat{S}_g(\bar{r},\bar{\Omega},t) = \frac{\int_{\Delta E_g} \phi^*(\bar{r},E,\bar{\Omega},t)S(\bar{r},E,\bar{\Omega},t)dE}{\hat{\phi}^*_g(\bar{r},\bar{\Omega},t)}$$

$$\hat{S}^*_g(\bar{r},\bar{\Omega},t) = \frac{\int_{\Delta E_g} S^*(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)dE}{\phi_g(\bar{r},\bar{\Omega},t)}$$

$$= \frac{\int_{\Delta E_g} P^{\phi}(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)dE}{\phi_g(\bar{r},\bar{\Omega},t)} = P^{\phi}_g(\bar{r},\bar{\Omega},t) \ .$$

However, as noted earlier, usually forward weighted group parameters are input to MORSE and the group adjoint fluxes $\phi^*_g(\bar{r},\bar{\Omega},t)$ are calculated. As a direct consequence of the derivation of the $\phi^*_g(\bar{r},\bar{\Omega},t)$ defining equation, Eq. (57), the effect of interest for the gth group is also given by

$$\lambda^*_g = \iiint \phi^*_g(\bar{r},\bar{\Omega},t)S_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \qquad \lambda = \sum_g \lambda^*_g$$

$$\lambda_g = \iiint \phi_g(\bar{r},\bar{\Omega},t)S^*_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \ , \qquad = \sum_g \lambda_g \qquad (70)$$

where

$\phi^*_g(\bar{r},\bar{\Omega},t)$ is the group adjoint flux corresponding to the forward weighted group parameters,

$$S_g(\bar{r},\bar{\Omega},t) = \int_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE \ ,$$

$$S^*_g(\bar{r},\bar{\Omega},t) = P^{\phi}_g(\bar{r},\bar{\Omega},t) \ . \qquad (71)$$

The derivation from this point on will implicitly assume forward weighted group parameters. However, the results can, with slight modification, be made to correspond to the adjoint weighted group parameters.

Substitution of Eq. (71) into Eq. (64) yields

$$S^*_{Tg}(\bar{r},\bar{\Omega},t) = \int dR\, e^{-\beta^*(\bar{r},R,\bar{\Omega})}\, P^o_g(\bar{r}',\bar{\Omega},t') \ , \tag{72}$$

and according to Equations (24) and (29), Eq. (72) can be rewritten as Equations (73) and (74), respectively:

$$S^*_{Tg}(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma^g_t(\bar{r}')\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})}\, P^{\psi}_g(\bar{r}',\bar{\Omega},t') \tag{73}$$

and

$$S^*_{Tg}(\bar{r},\bar{\Omega},t) = P^{\chi}_g(\bar{r},\bar{\Omega},t) \ . \tag{74}$$

Substitution of Eq. (73) into Eq. (67) and Eq. (74) into Eq. (67) yields the following forms for the "Integral Point-Value Equation:"

$$\chi^*_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}',\bar{\Omega})\,\{P^{\psi}_g(\bar{r}',\bar{\Omega},t') + C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\,\chi^*_{g'}(\bar{r}',\bar{\Omega}',t')\} \tag{75}$$

and

$$\chi^*_g(\bar{r},\bar{\Omega},t) = P^{\chi}_g(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\,\chi^*_{g'}(\bar{r}',\bar{\Omega}',t') \ . \tag{76}$$

Integral Event-Value Equation

At this point let us introduce a value function based on the event density and to relate this quantity to the point-value function by considering a particle leaving a collision at $\bar{r}$ with phase space coordinates (group g, $\bar{\Omega}$, time t). The value of this particle to the effect of interest is the point-value function $\chi^*_g(\bar{r},\bar{\Omega},t)$. This particle will experience an event in dR about $\bar{r}' = \bar{r} + R\bar{\Omega}$ with the probability $[\Sigma^g_t(\bar{r}')e^{-\beta^*_g(\bar{r},R,\bar{\Omega})}dR]$ and the value of this event (to the effect of interest) will be referred to as the "event-value" and be denoted by $W_g(\bar{r}',\bar{\Omega},t')$. That is, the "event-value" $W_g(\bar{r}',\bar{\Omega},t')$ is defined as the value (to the effect of interest) of having an event at $\bar{r}'$ with an incoming particle which has

phase space coordinates (group g, $\bar{\Omega}$, time t'). The sum of all such contributions to the effect of interest is given by

$$\int_0^\infty dR\ \Sigma_t^g(\bar{r}')\ e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\ W_g(\bar{r}',\bar{\Omega},t')\ ,$$

and, if the event-value function is properly defined, should equal the point-value function; that is,

$$\chi_g^*(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\ \Sigma_t^g(\bar{r}')\ e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\ W_g(\bar{r}',\bar{\Omega},t') \qquad (77)$$

or

$$\chi_g^*(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\rightarrow\bar{r}',\bar{\Omega})\ W_g(\bar{r}',\bar{\Omega},t')\ . \qquad (78)$$

A comparison of Eq. (78) with Eq. (75) would show that $W_g(\bar{r},\bar{\Omega},t)$ can be identified as

$$W_g(\bar{r},\bar{\Omega},t) = P_g^\psi(\bar{r},\bar{\Omega},t) + C_{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')\ \chi_{g'}^*(\bar{r},\bar{\Omega}',t)\ , \qquad (79)$$

and substitution of Eq. (78) into Eq. (79) yields the defining equation for the "Event-Value Function"

$$W_g(\bar{r},\bar{\Omega},t) = P_g^\psi(\bar{r},\bar{\Omega},t) + C_{g\rightarrow g'}(\bar{r},\bar{\Omega}\rightarrow\bar{\Omega}')\ T_{g'}(\bar{r}\rightarrow\bar{r}',\bar{\Omega}')\ W_{g'}(\bar{r}',\bar{\Omega}',t')\ . \qquad (80)$$

Equation (80) will be referred to as the "Integral Event-Value Equation." A comparison of Eq. (80) with Eq. (38) would show that the event-value function $W_g(\bar{r},\bar{\Omega},t)$ is adjoint to the event density $\psi_g(\bar{r},\bar{\Omega},t)$. Therefore the effect of interest is given by

$$\lambda = \sum_g \iiint S_c^g(\bar{r},\bar{\Omega},t)\ W_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt\ . \qquad (81)$$

Integral Emergent Adjuncton Density Equation

The solution of either the point-value equation, Eq. (76), or the event-value equation, Eq. (80), could be accomplished by Monte Carlo procedures; however, the random walk would not be the same as that implied by Eq. (40)*. Consider the following altered form of Eq. (76),

*The desire in MØRSE is to use the same random walk logic for both forward and adjoint calculations.

$$\chi_g^*(\bar{r},\bar{\Omega},t) = P_g^{\chi}(\bar{r},\bar{\Omega},t) + \int dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} \left[\frac{\Sigma_t^g(\bar{r}')}{\Sigma_t^g(\bar{r})}\right] \tag{82}$$

$$\times \sum_{g'} \int d\bar{\Omega}' \, \frac{\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\sum_g \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')} \left[\frac{\sum_g \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^g(\bar{r}')}\right] \chi_g^*(\bar{r}',\bar{\Omega}',t') \, .$$

The additional weight factor $[\Sigma_t^g(\bar{r}')/\Sigma_t^g(\bar{r})]$ arises since Eq. (76) and its altered form (Eq. (82), are actually <u>flux-like</u> equations, even though $\chi_g^*(\bar{r},\bar{\Omega},t)$ is adjoint to the emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$.

In a fashion analogous to the forward problem, the following new quantities are defined:

$$H_g(\bar{r},\bar{\Omega},t) \equiv \Sigma_t^g(\bar{r})\, \chi_g^*(\bar{r},\bar{\Omega},t) \tag{83}$$

and

$$H_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}',\bar{\Omega})\, G_g(\bar{r}',\bar{\Omega},t') \, . \tag{84}$$

Since $\chi_g^*(\bar{r},\bar{\Omega},t)$ is a flux-like variable, the new variable $H_g(\bar{r},\bar{\Omega},t)$ can be regarded as an event density and $G_g(\bar{r},\bar{\Omega},t)$ like an emergent particle density. The defining integral equation for $G_g(\bar{r},\bar{\Omega},t)$ should be the proper basis for an adjoint random walk.

The defining equation for the adjoint event density function $H_g(\bar{r},\bar{\Omega},t)$ is obtained by considering the following altered form of Eq. (75):

$$\chi_g^*(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma_t^g(\bar{r}')\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} [P_g^{\psi}(\bar{r}',\bar{\Omega},t') \tag{85}$$

$$+ C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\, \chi_{g'}(\bar{r}',\bar{\Omega}',t')] \, .$$

Multiply Eq. (85) by $\Sigma_t^g(\bar{r})$ and rearrange as follows:

$$\Sigma_t^g(\bar{r})\, \chi_g^*(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma_t^g(\bar{r})\, e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} [\Sigma_t^g(\bar{r}')\, P_g^{\psi}(\bar{r}',\bar{\Omega},t') \tag{86}$$

$$+ \check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\, \Sigma_t^{g'}(\bar{r}')\, \chi_{g'}^*(\bar{r}',\bar{\Omega}',t')]$$

where

$$\check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \equiv \frac{\Sigma_t^g(\bar{r}')}{\Sigma_t^{g'}(\bar{r})} C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \ . \tag{87}$$

Noting that:

$$H_g(\bar{r},\bar{\Omega},t) = \Sigma_t^g(\bar{r})\ \chi_g^*(\bar{r},\bar{\Omega},t) \ ,$$

$$\int dR\ \Sigma_t^g(\bar{r})\ e^{-\beta_g^*(\bar{r},R,\bar{\Omega})} = T_g(\bar{r}\to\bar{r}',\bar{\Omega}) \ ,$$

and

$$\Sigma_t^g(\bar{r})\ P_g^{\psi}(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) \ ,$$

Eq. (86) becomes

$$H_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}',\bar{\Omega})[P_g^{\phi}(\bar{r}',\bar{\Omega},t') + \check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')H_{g'}(\bar{r}',\bar{\Omega}',t')] \ . \tag{88}$$

A comparison of Eq. (88) with Eq. (84) reveals that

$$G_g(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) + \check{C}_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ H_{g'}(\bar{r},\bar{\Omega}',t) \ , \tag{89}$$

and the subsequent substitution of Eq. (84) into Eq. (89) yields the following defining equation for the adjoint emergent particle density:

$$G_g(\bar{r},\bar{\Omega},t) = P_g^{\phi}(\bar{r},\bar{\Omega},t) + \check{C}_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ T_{g'}(\bar{r}\to\bar{r}',\bar{\Omega}')\ G_{g'}(\bar{r}',\bar{\Omega}',t') \ . \tag{90}$$

Equation (90) is almost identical with Eq. (40) which defines the forward emergent particle density $\chi_g(\bar{r},\bar{\Omega},t)$ and also serves as the formal basis for the forward random walk. At this point, let us interpret Eq. (90) in terms of the transport of pseudo-particles called "adjunctons" in the (P'→P) direction of phase space. This presents two immediate problems:

1) The transport of the adjunctons from $\bar{r}' = \bar{r} + R\bar{\Omega}$ to $\bar{r}$ would be in a direction opposite to the direction vector $\bar{\Omega}$ -- therefore, the direction vector for the adjuncton should be $\hat{\Omega} \equiv -\bar{\Omega}$, and $\bar{r}' = \bar{r} - R\hat{\Omega}$.

2) The collision kernel should be interpreted as describing the (E'→E) change in phase space experienced by the adjuncton during its random walk; therefore, let

$$C_{g'\to g}(\bar{r},\hat{\Omega}'\to\hat{\Omega}) \equiv \check{C}_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}') = \sum_{g'} \int d\bar{\Omega}' \, \frac{\Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^{g'}(\bar{r})} \, . \qquad (91)$$

Equation (91) may be rewritten in terms of a normalized collision kernel and a weight factor:

$$C_{g'\to g}(\bar{r},\hat{\Omega}'\to\hat{\Omega}) = \sum_{g'} \int d\bar{\Omega}' \, \frac{\Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')}{\sum_{g} \int d\bar{\Omega} \, \Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')} \left\{ \frac{\sum_{g} \int d\bar{\Omega} \, \Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^{g'}(\bar{r})} \right\}^{*} . \qquad (92$$

The selection of new phase space coordinates (group g, $\hat{\Omega} = -\bar{\Omega}$) is made from the normalized kernel and the weight of the adjuncton is modified by the weight factor $[\,]^*$ which is no longer a simple non-absorption probability and may assume values in excess of unity. Therefore, there is no "analogue" scattering for adjunctons and the adjuncton's weight may increase at some collisions.

Equation (90) can be rewritten as

$$G_g(\bar{r},\hat{\Omega},t) = P_g^{\phi}(\bar{r},\hat{\Omega},t) + C_{g'\to g}(\bar{r},\hat{\Omega}'\to\hat{\Omega}) \; T_{g'}(\bar{r}'\to\bar{r},\hat{\Omega}') \; G_{g'}(\bar{r}',\hat{\Omega}',t') \; , \qquad (93)$$

which now corresponds to the transport of adjunctons and provides the desired basis for the adjoint random walk in the MØRSE code. Note that the source of adjunctons is provided by $P_g^{\phi}(\bar{r},\hat{\Omega},t)$ which is related to $P_g^{\phi}(\bar{r},\bar{\Omega},t)$ as follows:

$$P_g^{\phi}(\bar{r},\hat{\Omega},t) = P_g^{\phi}(\bar{r},-\bar{\Omega},t) \; , \qquad (94)$$

which must be taken into consideration if the response function $P_g^{\phi}(\bar{r},\bar{\Omega},t)$ has angular dependence -- however, many physical situations permit an isotropic assumption for the $\bar{\Omega}$-dependence.

A Monte Carlo solution of Eq. (93), the "integral emergent adjuncton dens equation," will generate data from which the adjuncton flux $\chi_g^*(\bar{r},\hat{\Omega})$ and

other quantities of interest can be determined. The general use of $\chi_g^*(\bar{r},\hat{\Omega})$ must take into account the reversal of direction between adjunctons and real particles, i.e., $\hat{\Omega} = -\bar{\Omega}$. For example, consider the various ways of calculating the answer of interest:

$$\lambda = \sum_g \iiint P_g^{\phi}(\bar{r},\bar{\Omega},t)\ \phi_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{95}$$

$$= \sum_g \iiint \frac{P_g^{\phi}(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\to\bar{r},\bar{\Omega})\ \chi_g(\bar{r}',\bar{\Omega},t')d\bar{r}d\bar{\Omega}dt$$

$$\lambda = \sum_g \iiint S_g(\bar{r},\bar{\Omega},t)\chi_g^*(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint S_g(\bar{r},\bar{\Omega},t)\chi_g^*(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{96}$$

$$\lambda = \sum_g \iiint S_c^g(\bar{r},\bar{\Omega},t)W_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint S_c^g(\bar{r},\bar{\Omega},t)W_g(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{97}$$

$$\lambda = \sum_g \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} H_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} H_g(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{98}$$

$$\lambda = \sum_g \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\to\bar{r},\bar{\Omega})G_g(\bar{r}',\bar{\Omega},t')d\bar{r}d\bar{\Omega}dt$$

$$= \sum_g \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\to\bar{r},-\hat{\Omega})\ G_g(\bar{r}',-\hat{\Omega},t')d\bar{r}d\bar{\Omega}dt\ . \tag{99}$$

Further, if outward boundary crossings would be scored in the forward problem, the corresponding source adjunctons would be introduced in the inward direction. Likewise, adjunctons would be scored for entering a volume from which the source particles in the forward problem would be emitted. It should be noted that many sources and response functions are isotropic and the problem of direction reversal need not be considered.

Multiplying Systems

The general integral equations in group notation of the previous section are here specialized to the problem of multiplying systems. In a fissioning system it will be presumed that the source of neutrons for the nth generation comes from fissions which occur during the previous generation, the (n-1)st generation. In group notation and seven-dimensional phase space, the source term for the nth generation, $S_g^n(\bar{r},\bar{\Omega},t)$, is given by

$$S_g^n(\bar{r},\bar{\Omega},t) = \int_{\Delta E_g} S^n(\bar{r},E,\bar{\Omega},t)dE \qquad (100)$$

where

$S^n(\bar{r},E,\bar{\Omega},t)dEd\bar{\Omega}$ = source particles emitted for the nth generation per unit volume and time at the space point $\bar{r}$ and time t with energies in dE about E and directions which lie in $d\bar{\Omega}$ about $\bar{\Omega}$,

$$S^n(\bar{r},E,\bar{\Omega},t) = \frac{f(E)}{4\pi} \iint_{4\pi} dE'd\bar{\Omega}' \; \nu\Sigma_f(\bar{r},E')\phi^{n-1}(\bar{r},E',\bar{\Omega}',t) \qquad (101)$$

$f(E)dE$ = fraction of fission neutrons emitted having energies in dE about E,

$\phi^{n-1}(\bar{r},E,\bar{\Omega},t)$ = angular neutron flux for the (n-1)st generation,

$\nu\Sigma_f(\bar{r},E')$ = fission neutron yield x macroscopic fission cross section.

Substitution of Eq. (101) into Eq. (100) and expressing the energy integration as a summation over energy groups yields

$$S_g^n(\bar{r},\bar{\Omega},t) = \frac{f_g}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \; \nu\Sigma_f^{g'}(\bar{r}) \; \phi_{g'}^{n-1}(\bar{r},\bar{\Omega}',t) \; , \qquad (102)$$

---

* The terms generation and batch will be used interchangeably in this section and will refer to the batches of neutrons processed in the MØRSE Monte Carlo calculation.

where

$$f_g = \int_{\Delta E_g} f(E)dE \tag{103}$$

$$\nu\Sigma_f^g(\bar{r}) = \frac{\int_{\Delta E_g} \nu\Sigma_f(\bar{r},E)\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g} \phi(\bar{r},E,\bar{\Omega},t)dE} \tag{104}$$

Equation (102) can also be expressed in terms of the emergent particle density:

$$S_g^n(\bar{r},\bar{\Omega},t) = \frac{f_g}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})} \int_0^\infty dR\, \Sigma_t^{g'}(\bar{r}) e^{-\beta_g(\bar{r},R,\bar{\Omega}')} \chi_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') \tag{105}$$

where

$$\phi_{g'}(\bar{r},\bar{\Omega}',t) = \int_0^\infty dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t') \quad , \tag{106}$$

so that for a given $S_g^n(\bar{r},\bar{\Omega},t)$, the emergent particle density distribution for the nth generation can be calculated using the following modified form of Eq. (27):

$$\chi_g^n(\bar{r},\bar{\Omega},t) = S_g^n(\bar{r},\bar{\Omega},t)$$

$$+ \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}' \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \int_0^\infty dR\, \Sigma_t^{g'}(\bar{r})\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}^n(\bar{r}',\bar{\Omega}',t')$$

$$= S_g^n(\bar{r},\bar{\Omega},t) + C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\, T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}')\, \chi_{g'}^n(\bar{r}',\bar{\Omega}',t') \quad . \tag{107}$$

Equations (105) and (107) can be combined and written as an eigenvalue equation in seven-dimensional phase space

$$\chi_g(\bar{r},\bar{\Omega},t) = \frac{1}{k}\frac{f_g}{4\pi}\sum_{g'=G}^{1}\int_{4\pi} d\bar{\Omega}'\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\int dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_g(\bar{r},R,\bar{\Omega})}\chi_{g'}(\bar{r}',\bar{\Omega},t')$$

$$+\sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})}\int_0^{\infty} dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\chi_{g'}(\bar{r}',\bar{\Omega}',t'). \tag{108}$$

The usual objective in a reactor calculation is to find the eigenfunctions $\chi_g(\bar{r},\bar{\Omega},t)$, and the eigenvalue k. In MORSF this is accomplished iteratively, each batch being one iteration. The source for the first batch is unknown and must be assumed. From this source an estimate of the resulting emergent particle densities, χ_g^1, are calculated from Eq. (107). The source for the next batch, S_g^2, is obtained from Eq. (105) and then estimates of the χ_g^2 are obtained from Eq. (107). After the source has converged (usually after a few batches), the χ_g^n are presumed to be a valid estimate of the eigenfunction χ_g in Eq. (108) and an estimate of the multiplication factor can be obtained for each of the succeeding batches.

The multiplication factor corresponding to the nth generation (or batch) is defined as the ratio of the total production of fission neutrons during the nth generation to the total number of source neutrons introduced into the nth generation

$$k_n = \frac{\sum_{g'=G}^{1}\iiint d\bar{r}d\bar{\Omega}'dt\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\int_0^{\infty} dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\chi_{g'}^n(\bar{r}',\bar{\Omega}',t')}{\sum_{g'=G}^{1}\iiint S_{g'}^n\, d\bar{r}d\bar{\Omega}dt} \tag{109}$$

which can also be expressed as the ratio of successive sources

$$k_n = \frac{\sum_{g'=G}^{1}\iiint S_{g'}^{n+1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt}{\sum_{g'=G}^{1}\iiint S_{g'}^{n}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt}. \tag{110}$$

The multiplication factor is calculated at the end of each batch and the eigenvalue, k, is taken as the mean value of the k_n averaged over all the batches calculated after convergence of the eigenfunctions was achieved.

Equation (107) is solved by MØRSE in the same manner as it would be for non-fissioning systems. The fission event is treated as an absorption and the neutron's weight is modified accordingly, i.e., fissions that occur do not introduce new neutrons into the present generation. The multiplication factor, k_n, is estimated by summing the contribution $\nu\Sigma_f^g(\bar{r})/\Sigma_t^g(\bar{r})$. W_b at every collision (W_b, the neutron's weight before collision, is an estimate of the collision density). At the end of the batch, k_n is divided by N, the total starting weight of the batch.

The source for the next batch is not obtained directly from the individual contributions ($\nu\Sigma_f^g \cdot W_b$). Rather, Russian roulette and splitting are used to discretize these contributions into ones of equal value. The splitting and Russian roulette parameters used are determined by the input parameter, FWLØW, the desired value of a single contribution. To keep the number of neutrons from multiplying or decreasing indefinitely, FWLØ is modified from batch to batch such that the number of source neutrons for each batch remains nearly constant. The value of FWLØW for the (n+1)st batch is calculated at the completion of the nth batch as follows:

$$FWL\emptyset_{n+1} = FWL\emptyset_n \cdot \bar{k}_n \cdot \frac{\text{(fission neutrons produced during the nth batch)}}{\text{(source neutrons introduced into the first batch)}} , \quad (111)$$

where $\bar{k}_n$ is an accumulative estimate of k through n batches. The $\bar{k}_n$ modifying factor is required since the FWLØ calculated after the nth batch affects the number of source neutrons in the (n+2)nd batch.

The adjoint problem for the fissioning system is solved by MØRSE in terms of the random walk of "adjunctons" as described by the integral emergent adjuncton density equation, Eq. (93), that can be rewritten in batch notation as

$$G_g^n(\bar{r},\hat{\Omega},t) = [P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n + C_{g'\to g}(\bar{r},\hat{\Omega}'\to\hat{\Omega})\, T_{g'}(\bar{r}'\to\bar{r},\hat{\Omega}')\, G_{g'}^n(\bar{r}',\hat{\Omega}',t'), \quad (112)$$

with the source of adjunctons being provided by the response function based on flux density, P_g^{ϕ}. The effect of interest for the nth generation, λ_g^n, is the production of fission neutrons due to fissions in group g that appear at the fission site in the next generation according to the group fission spectrum, f_g, and is given by

$$\lambda_g^{n-1} = \iiint [P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n \phi_g^{n-1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{113}$$

$$= \iiint \left(\nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi} \chi_{g'}^{*n-1}(\bar{r},\bar{\Omega}',t) \right) \phi_g^{n-1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt$$

where

$\chi_g^{*n}(\bar{r},\bar{\Omega},t)$ = the value to the effect of interest in the nth generation of an emergent neutron with phase space coordinates (group g, $\bar{r}$, $\bar{\Omega}$,t).

From Eq. (113), the source of adjunctons for the nth generation is identified as

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n = \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi} \chi_{g'}^{*n-1}(\bar{r},\bar{\Omega}',t) . \tag{114}$$

Noting that according to Equations (83) and (84)

$$\chi_g^{*n}(\bar{r},\bar{\Omega},t) = \frac{1}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\to\bar{r},\bar{\Omega}) G_g^n(\bar{r}',\bar{\Omega},t') , \tag{115}$$

Eq. (114) can be rewritten as

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n = \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi \Sigma_t^{g'}(\bar{r})} T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}') G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t')$$

$$= \frac{\nu\Sigma_f^g(\bar{r})}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' f_{g'} \int_0^{\infty} dR\, e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') . \tag{116}$$

Noting that the fission process is independent of the incident neutron's direction and that the fission-neutrons are emitted isotropically

$$[P_g^{\phi}(\bar{r},\bar{\Omega},t)]^n \equiv [P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n \quad , \tag{117}$$

and Eq. (116) can be used in conjunction with Eq. (112), i.e., $\bar{\Omega}$ replaced by $\hat{\Omega}$.

Equation (116) can be rewritten as

$$[P_g^{\phi}(\bar{r},\hat{\Omega},t)]^n = \frac{1}{4\pi}\frac{\nu\Sigma_f^g(\bar{r})}{\nu\Sigma_f(\bar{r})}\sum_{g'=G}^{1}\int_{4\pi} d\hat{\Omega}'\,[f_{g'}\,\nu\Sigma_f(\bar{r})]\int_0^{\infty} dR\; e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')}$$

$$\times\, G_{g'}^{n-1}(\bar{r}',\hat{\Omega}',t') \quad , \tag{118}$$

where

$$\nu\Sigma_f(\bar{r}) = \sum_{g=G}^{1}\nu\Sigma_f^g(\bar{r}), \tag{119}$$

$\frac{\nu\Sigma_f^g(\bar{r})}{\nu\Sigma_f(\bar{r})}$ = energy distribution of adjunctons emerging from an adjoint fission,

$[f_{g'}\nu\Sigma_f(\bar{r})]$ = the g'th group cross section for adjoint fission.

It is noted that the integral emergent particle density equation, Eq. (107), is identical in form with the integral emergent adjuncton density equation, Eq. (112), so that essentially* the same random walk procedures can apply to the solution of the forward and adjoint fissioning systems. The adjoint source, Eq. (118), differs from the forward source, Eq. (105), only in that the fission cross section and the group fission spectrum have changed their roles. The adjoint-fission group cross section is $[f_{g'}\nu\Sigma_f(\bar{r})]$ and the energy distribution of the adjunctons emerging from an adjoint fission is $\nu\Sigma_f^g(\bar{r})/\nu\Sigma_f(\bar{r})$.

---

*The same differences will exist between the forward and adjoint collision kernels here as was the case for non-fissioning systems.

The adjoint solution is started by assuming some arbitrary initial source, $[P_g^{\phi}(\bar{r},\hat{\Omega},t)]^1$, and calculating the $G_g^1(\bar{r},\hat{\Omega},t)$ using Eq.(112). A new source term, $[P_g^{\phi}(\bar{r},\hat{\Omega})]^2$ is then calculated from Eq. (118) and the next estimate, $G_g^2(\bar{r},\hat{\Omega})$ is calculated using Eq. (112). This procedure continues until, as in the forward case, the source has converged and the G_g^n's are presumed to be an estimate of the eigenfunctions G_g. Then for each succeeding batch, the following estimate is made for the eigenvalue k:

$$k_n = \frac{\sum_{g'=G}^{1} \iiint d\bar{r}\,d\hat{\Omega}'\,dt\,[f_{g'} \cdot \nu\Sigma_f^{g'}(\bar{r})] \int_0 dR\; e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')}\, G_{g'}^n(\bar{r}',\hat{\Omega}',t')}{\sum_{g'=G}^{1} \iiint [P_{g'}^{\phi}(\bar{r},\hat{\Omega},t)]^n d\bar{r}\,d\hat{\Omega}\,dt} \quad , (120$$

and the eigenvalue, k, is taken as the mean value of the k_n averaged over all the batches calculated after convergence of the eigenfunctions was achieved -- exactly the same procedure used in the forward calculation.

4.11 Generalized Gaussian Quadrature

General Statement of the Problem and Its Solution

Given $\omega(x)$, $a \leq x \leq b$, such that $\omega(x) \geq 0$ (Restriction I).

Problem: find $\{x_i, \omega_i\}$ for $i = 1,n$ so that:

$$\int_a^b f(x)\,\omega(x)\,dx = \sum_{i=1}^{n} f(x_i)\cdot\omega_i \quad \text{(Restriction II)}$$

holds for all $f(x)$ where $f(x)$ is a polynomial of degree $2n-1$ or less.

Solution: Determine a set of polynomials $Q_i(x)$ $(i=1,n)$ orthogonal with respect to $\omega(x)$. That is

$$\int_a^b Q_i(x)\,Q_j(x)\,\omega(x)\,dx = \delta_{ij}\,N_i \quad ,$$

where δ_{ij} is the Kronecker delta and N_i is a normalization constant. Then $\{x_i\}_{i=1}^{n}$ are given by the roots of $Q_n(x)$, $Q_n(x_i) = 0$, and

$$\omega_i = \left[\sum_{i=1}^{n-1} Q_i^2(x_i)/N_i\right]^{-1} .$$

Note: Since the functions $1,x,x^2,\ldots,x^{2n-1}$ are independent and form a basis for the space of all polynomials of degree $2n-1$ or less, it is equivalent to Restriction II to require that

$$M_\nu = \int_a^b x^\nu\,\omega(x)\,dx = \sum_{i=1}^{n} x_i^\nu\cdot\omega_i \quad \text{for } \nu = 0,2n-1 .$$

In other words, the problem is that of finding a discrete distribution.

$$\omega^*(x) = \sum_{i=1}^{n} \omega_i\,\delta(x - x_i) \quad ,$$

having its first 2n moments, $\{M_\nu\}_{\nu=0}^{2n-1}$ identical to those of the original distribution $\omega(x)$.

It is then possible to relax the non-negativity restriction, $\omega(x) \geq 0$, and in fact to state that $\omega(x)$ need not be completely specified but only its first 2n moments be given. Restriction I then becomes two restrictions on the moments:

$$I_a: \qquad |C_i| \geq 0 \quad i=1, n-1$$

where $|C_i|$ is the Gram determinant

$$|C_1| = \begin{vmatrix} M_0 M_1 \\ M_1 M_2 \end{vmatrix}, \quad |C_2| = \begin{vmatrix} M_0 M_1 M_2 \\ M_1 M_2 M_3 \\ M_2 M_3 M_4 \end{vmatrix}, \quad \ldots, \quad |C_{n-1}| = \begin{vmatrix} M_0 M_1 M_2 \cdots M_{n-1} \\ M_1 M_2 \quad \cdots M_n \\ \cdot \\ \cdot \\ \cdot \\ M_{n-1} M_n \text{ --- } M_{2n-2} \end{vmatrix}$$

and

I_b) The roots of $Q_n(x)$ lie inside the interval [a,b], i.e., $a \leq x_i \leq b$ whenever $Q_n(x_i) = 0$.

Equivalence of Moments and Legendre Coefficients

We shall use the following form for the Legendre expansion of an angular distribution:

$$f(\mu) = \sum_{\ell=0}^{\infty} \frac{2\ell + 1}{2} f_\ell P_\ell(\mu) . \tag{1}$$

From this it follows that

$$f_\ell = \int_{-1}^{1} f(\mu) P_\ell(\mu) \, d\mu \text{ and } f_0 \equiv 1 . \tag{2}$$

The moments of the distribution are defined by

$$M_n = \int_{-1}^{1} \mu^n f(\mu) \, d\mu . \tag{3}$$

If the Legendre polynomials are written

$$P_\ell(\mu) = \sum_{n=0}^{\ell} p_{\ell n}\, \mu^n \quad , \tag{4}$$

[the $p_{\ell n}$'s may be derived easily from the recurrence relation for $P_\ell(\mu)$]. Then it follows simply from Equation (2) that

$$f_\ell = \sum_{n=0}^{\ell} p_{\ell n} \int_{-1}^{1} f(\mu)\, \mu^n\, d\mu = \sum_{n=0}^{\ell} p_{\ell n}\, M_n \quad . \tag{5}$$

Likewise

$$M_n = \int_{-1}^{1} \mu^n\, f(\mu)\, d\mu = \sum_{\ell=0}^{\infty} \frac{2\ell + 1}{2}\, f_\ell \int_{-1}^{1} \mu^n\, P_\ell(\mu)\, d\mu \quad .$$

From the orthogonality property we know that $P_\ell(\mu)$ is orthogonal to any polynomial of degree less than ℓ. Hence

$$\int_{-1}^{1} \mu^n\, P_\ell(\mu)\, d\mu = 0 \quad \text{for } \ell > n \ .$$

Then

$$M_n = \sum_{\ell=0}^{n} \frac{2\ell + 1}{2}\, f_\ell\, p_{n\ell}^{-1} \tag{6}$$

where

$$p_{n\ell}^{-1} = \int_{-1}^{1} \mu^n\, P_\ell(\mu)\, d\mu$$

are the coefficients for a Legendre expansion of μ^n, that is,

$$\mu^n = \sum_{\ell=0}^{n} \frac{2\ell + 1}{2}\, p_{n\ell}^{-1}\, P_\ell(\mu) \quad .$$

[The Legendre polynomial recurrence relation may also be used to derive recurrence relations for the $p_{n\ell}^{-1}$.]

Equations (5) and (6) show that the first n moments of an angular distribution may be derived from the first n Legendre coefficients and vice versa.

Generation of Polynomials Orthogonal With Respect to $\omega(x)$

Let us now presume that we are given the first 2n moments, $M_0, M_1, \ldots, M_{2n-1}$, of an arbitrary function $\omega(x)$ and are given no additional information about $\omega(x)$. We shall attempt to derive a set of polynomials which are orthogonal with respect to $\omega(x)$. If we define the notation

$$E[I(x)] = \int_a^b I(x)\, \omega(x)\, dx ,$$

then what we wish is to determine $Q_0, Q_1, \ldots, Q_n$ such that

$$Q_i(x) = \sum_{k=0}^{i} a_{ik}\, x^k , \tag{7}$$

with the normalization condition $a_{ii} = 1$, and that

$$E[Q_i(x)\, Q_j(x)] = \delta_{ij}\, N_i . \tag{8}$$

Note that

$$N_i = E[Q_i^2(x)] = \int_a^b Q_i^2(x)\, \omega(x)\, dx .$$

Since $\omega(x) \geq 0$, then it follows that*

$$N_i > 0 . \tag{9}$$

From the properties of orthogonal polynomials we know that an arbitrary polynomial of order i, $S_i(x)$, may be expanded in terms of the Q polynomials,

---

*Since we wish to relax the non-negativity restriction slightly but not completely, we will retain Eq. (9) as a reasonable requirement for a "well-behaved" $\omega(x)$. This requirement is essential to allow full use of the properties of orthogonal polynomials. It is also essential to the eventual use of this development as a Monte Carlo selection technique since it is needed to ensure that the "probabilities," ω_i, be positive.

$$S_i(x) = \sum_{k=0}^{i} s_{ik} Q_k(x) \quad .$$

It follows that

$$E[S_i(x)\, Q_j(x)] = 0 \quad \text{for } i < j \ .$$

Let us presume that we have obtained the first i polynomials and are attempting to derive $Q_{i+1}(x)$. Due to our normalization condition ($a_{ii} = 1$) we have

$$Q_{i+1}(x) = x^{i+1} + R_i(x) \ , \tag{10}$$

where

$$R_i(x) = \sum_{k=0}^{i} a_{i+1,k}\, x^k \ .$$

$$\begin{aligned} Q_{i+1}(x) &= x \cdot x^i + R_i(x) \\ &= x \cdot [Q_i(x) - R_{i-1}(x)] + R_i(x) \\ &= x\, Q_i(x) + [R_i(x) - x\, R_{i-1}(x)] \ . \end{aligned}$$

The term $R_i(x) - x\, R_{i-1}(x)$ is a polynomial of order i and may be expanded in terms of the Q's. Thus

$$Q_{i+1}(x) = x\, Q_i(x) + \sum_{k=0}^{i} d_{ik}\, Q_k(x) \ . \tag{11}$$

For $j \leq i-2$ we can use the orthogonality relation

$$\begin{aligned} E[Q_{i+1}(x)\, Q_j(x)] = 0 &= E[x\, Q_i(x)\, Q_j(x)] + \sum_{k=0}^{i} d_{ik}\, E[Q_k(x)\, Q_j(x)] \\ &= E[Q_i(x)\, (x\, Q_j(x))] + d_{ij}\, N_j \\ &= d_{ij}\, N_j \ , \end{aligned}$$

since $x\,Q_j(x)$ is a polynomial of order $\leq i-1$ and is orthogonal to $Q_i(x)$. Since $N_j > 0$ we must have $d_{ij} = 0$.

If we write

$$\mu_{i+1} = - d_{i,i}$$

and

$$\sigma_i^2 = - d_{i,i-1} ,$$

then Eq. (11) reduces to

$$Q_{i+1}(x) = (x - \mu_{i+1})\, Q_i(x) - \sigma_i^2\, Q_{i-1}(x) . \qquad (12)$$

This equation is the basic recurrence relation for our polynomials. We have

$$E[Q_{i+1}(x)\, Q_{i-1}(x)] = 0$$

$$= E[x\, Q_i(x)\, Q_{i-1}(x)] - \mu_{i+1}\, E[Q_i(x)\, Q_{i-1}(x)] - \sigma_i^2\, E[Q_{i-1}^2(x)]$$

$$= E[Q_i(x)\, (x\, Q_{i-1}(x))] - \sigma_i^2\, N_{i-1}$$

$$= E[Q_i(x)\, \{Q_i(x) - \sum_{k=0}^{i-1} d_{i-1,k}\, Q_k(x)\}] - \sigma_i^2\, N_{i-1}$$

$$= E[Q_i^2(x)] - \sigma_i^2\, N_{i-1}$$

$$= N_i - \sigma_i^2\, N_{i-1} .$$

This is easily solved for

$$\sigma_i^2 = N_i / N_{i-1} . \qquad (13)$$

If we return to Equation (10), it is easy to see that

$$N_i = E[Q_i(x)\, Q_i(x)] = E[Q_i(x)\, x^i] + E[Q_i(x)\, R_{i-1}(x)]$$

$$= E[Q_i(x)\, x^i] = \sum_{k=0}^{i} a_{ik} \int_a^b x^k\, x^i\, dx = \sum_{k=0}^{i} a_{ik}\, M_{k+i} . \qquad (14)$$

Likewise we will define

$$L_{i+1} = E[Q_i(x)\, x^{i+1}]$$

$$= \sum_{k=0}^{i} a_{i,k}\, M_{k+i+1} \quad . \tag{15}$$

Then the final orthogonality relation used in defining $Q_{i+1}(x)$ gives us

$$E[Q_{i+1}(x)\, Q_i(x)] = 0$$

$$= E[Q_{i+1}(x)\, x^i] + E[Q_{i+1}(x)\, R_{i-1}(x)]$$

$$= E[x\, Q_i(x)\, x^i] - \mu_{i+1}\, E[Q_i(x)\, x^i] - \sigma_i^2\, E[Q_{i-1}(x)\, x^i]$$

$$= L_{i+1} - \mu_{i+1}\, N_i - \sigma_i^2\, L_i$$

or

$$\mu_{i+1} = \frac{L_{i+1}}{N_i} - \sigma_i^2 \frac{L_i}{N_i}$$

$$= \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}} \quad . \tag{16}$$

The coefficients a_{ik} may be obtained from the recurrence relation, Eq. (12), by taking the coefficient of x^k on both sides of the equation. This gives

$$a_{i+1,k} = a_{i,k-1} - \mu_{i+1}\, a_{i,k} - \sigma_i^2\, a_{i-1,k} \quad . \tag{17}$$

To recapitulate, one uses moments through M_{2i} and the values of a_{ik} from $Q_i(x)$ to calculate N_i (Eq. 14). N_i, along with the previously determined N_{i-1}, allows one to calculate σ_i^2 (Eq. 13). The moments through M_{2i+1} and $Q_i(x)$ determine L_{i+1} (Eq. 15). This in turn allows the calculation of μ_{i+1} (Eq. 16). With σ_i^2 and μ_{i+1} the recurrence relation (Eq. 12) determines $Q_{i+1}(x)$. In sum the moments $M_0, M_1, \ldots, M_{2n-1}$ of $\omega(x)$ allow the determination of the orthogonal polynomials $Q_0(x)$, $Q_1(x)$, ..., $Q_n(x)$.

This is subject only to the restriction $N_i > 0$, $i = 0,n$. Although it is far from obvious, this restriction may be written in simple closed form as:

$$\begin{vmatrix} M_0 M_1 M_2 \cdots M_i \\ M_1 M_2 M_3 \cdots M_{i+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ M_i \; M_{i+1} \cdots M_{2i} \end{vmatrix} > 0 .$$

Properties of the Roots of the Orthogonal Polynomials

The roots of the orthogonal polynomials have two useful properties which we shall prove.

Lemma I: $Q_n(x)$ has n distinct, real roots which "interleave" with the roots of $Q_{n-1}(x)$; that is, between any two adjacent roots of $Q_{n-1}(x)$ there is one and only one root of $Q_n(x)$, and furthermore there is one root of $Q_n(x)$ greater than the largest root of $Q_{n-1}(x)$ and one smaller than the least root of $Q_{n-1}(x)$. Likewise there is one and only one root of $Q_{n-1}(x)$ between any two adjacent roots of $Q_n(x)$.

Proof: We assume the Lemma to be true for Q_{n-1} and Q_{n-2}. Let $x_1 > x_2 > \cdots > x_{n-1}$ be the roots of Q_{n-1}. Then it follows that the sequence $Q_{n-2}(x_1)$, $Q_{n-2}(x_2)$, ..., $Q_{n-2}(x_{n-1})$ alternates in sign. Since

$$Q_n(x_i) = (x_i - \mu_n)\, Q_{n-1}(x_i) - \sigma^2_{n-1}\, Q_{n-2}(x_i)$$

$$= - \sigma^2_{n-1}\, Q_{n-2}(x_i) .$$

The sequence $Q_n(x_i)$, $Q_n(x_2)$, ..., $Q_n(x_{n-1})$ also alternates in sign. This establishes that there is at least one root of Q_n between any two roots of Q_{n-1}. Because the Q_i's are normalized to $a_{ii} = 1$, they are all positive at $+\infty$ and alternate in sign at $-\infty$. Q_{n-2} has no root between x_1 and $+\infty$; hence $Q_{n-2}(x_1) > 0$. But $\sigma^2_{n-1} > 0$ (because $N_{n-1} > 0$ and $N_{n-2} > 0$); therefore, $Q_n(x_1) < 0$ and Q_n must have at least one root greater than x_1.

Similar reasoning leads to the conclusion that $Q_{n-2}(x_{n-1})$, $Q_{n-2}(x \to -\infty)$, and $Q_n(x \to -\infty)$ have the same sign while $Q_n(x_{n-1})$ is of the opposite sign. Thus Q_n must have at least one root between x_{n-1} and $-\infty$. Since this gives us n intervals where Q_n must have "at least one" root, it is clear that Q_n has n distinct roots which interleave with the roots of Q_{n-1}

The proof by induction may be completed by using similar arguments to show that one of the two roots of $Q_2(x)$ lies above the single root of $Q_1(x)$ and one below it.

Lemma II: The n roots of $Q_n(x)$ lie in the interval (a,b).

Proof: Assume that $Q_n(x)$ has only s changes of sign in the interval (a,b) at the points $x_1, x_2, \ldots, x_s$. Let

$$\theta(x) = (x - x_1)(x - x_2)(x - x_3) \ldots (x - x_s) ,$$

then $\theta(x)\, Q_n(x)$ does not change sign in the interval (a,b). It follows that*

$$E[\theta(x)\, Q_n(x)] = \int_a^b \theta(x)\, Q_n(x)\, \omega(x)\, dx \neq 0 .$$

However, $\theta(x)$ is a polynomial of order $s \leq n$. Since $Q_n(x)$ is orthogonal to all polynomials of order less than n, we must have $s = n$, thus proving the assertion.

The Meaning of the Two Restrictions Which Replace the Non-Negativity Requirement, $\omega(x) \geq 0$

In the foregoing development, knowledge of the entire function $\omega(x)$ is never required. Instead, all that is needed are the moments, $M_0, M_1, \ldots, M_{2n-1}$, of $\omega(x)$. The generalized quadrature thus developed is thereby valid for the whole class of functions having those moments. Since the moments are equivalent to the Legendre coefficients, $f_0, f_1, \ldots, f_{2n-1}$, this class is comprised of all functions having the same truncated

*This step relies on the requirement that $\omega(x)$ be non-negative. We wish to relax this restriction somewhat but not completely. Since Lemma II expresses a property which will be essential to the use of this development as a Monte Carlo selection technique, we will use this property as one of the requirements for a "well-behaved" $\omega(x)$ with which we shall replace the non-negativity restriction.

Legendre expansion; that is,

$$\omega(x) \approx \omega^*(x) = \sum_{\ell=0}^{2n-1} \frac{2\ell + 1}{2} f_\ell P_\ell(x) \quad .$$

In particular, the discrete distribution derived by this technique is itself one function from this class.

It is not required that all functions of this class be non-negative; in fact, there are infinitely many which are not. It is not even required that the truncated Legendre expansion $\omega^*(x)$ be non-negative. However, it is essential that at least one function in this class be non-negative. The restrictions

1) $N_i > 0$, $i = 1, \ldots, n$ and

2) $Q_n(x)$ has n roots in the interval $(-1, +1)$

express exactly this requirement. Then it follows that $\omega^*(x)$ is the truncated expansion of some unspecified non-negative function. The failure of either of those two conditions expresses the fact that the given moments (or Legendre coefficients) are not those of any everywhere positive function.

Generation of the Generalized Gaussian Quadrature

We are given $\omega(x)$, $a \leq x \leq b$, [or rather, we are given the moments of $\omega(x)$] and we are attempting to find a set of points, x_i, and associated weights, ω_i, so that, for any arbitrary polynomial, $f(x)$, of order 2n-1 or less,

$$E[f(x)] = \int_a^b f(x)\, \omega(x)\, dx = \sum_{i=1}^{n} f(x_i) \cdot \omega_i \quad .$$

By simple division of polynomials,

$$f(x) = q_{n-1}(x)\, Q_n(x) + r_{n-1}(x)$$

where $q_{n-1}(x)$ and $r_{n-1}(x)$ are polynomials of order n-1 or less.

$$E[f(x)] = E[q_{n-1}(x)\, Q_n(x)] + E[r_{n-1}(x)] \qquad (18)$$

$$= E[r_{n-1}(x)] \text{ from the orthogonality property of } Q_n.$$

However, we want

$$E[f(x)] = \sum_{i=1}^{n} f(x_i)\cdot\omega_i = \sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i)\cdot\omega_i + \sum_{i=1}^{n} r_{n-1}(x_i)\cdot\omega_i$$

$$= \sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i)\cdot\omega_i + E[r_{n-1}(x)] \,. \tag{19}$$

By subtracting Eq. (18) from Eq. (19), we find that we must require, for all polynomials, $q_{n-1}(x)$, that

$$\sum_{i=1}^{n} q_{n-1}(x_i)\, Q_n(x_i)\cdot\omega_i = 0 \,. \tag{20}$$

This condition can only be met if

$Q_n(x_i) = 0$, that is, the desired points, x_i, are the roots of $Q_n(x)$. (21)

Now we still must pick the weights, ω_i, so that

$$E[r_{n-1}(x)] = \sum_{i=1}^{n} r_{n-1}(x_i)\cdot\omega_i$$

where $r_{n-1}(x)$ is an arbitrary polynomial of order n-1 or less. Since r_{n-1} may be expanded as a linear sum of the orthogonal polynomials, Q_0, Q_1, ..., Q_{n-1}, it is sufficient to require

$$E[Q_k(x)] = \sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i \quad \text{for } k = 0,1,\ldots,n-1 \,. \tag{22}$$

However,

$$E[Q_k(x)] = E[Q_k(x)\, Q_o(x)] = N_o\, \delta_{ko} \,.$$

Thus we must have

$$\sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i = N_o\, \delta_{ko} \quad \text{for } k = 0,1,\ldots,n-1 \,. \tag{23}$$

Multiplying Eq. (23) by $[Q_k(x_j)/N_j]$ and summing over k, we find

$$\sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} \sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i = \sum_{i=1}^{n} \omega_i \Big\{ \sum_{k=0}^{n-1} \frac{Q_k(x_j)\, Q_k(x_i)}{N_k} \Big\} \tag{24}$$

$$= \sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} N_o\, \delta_{ko} = \frac{Q_o(x_j)}{N_o} N_o = 1 .$$

Introducing the function

$$D_{n-1}(x,y) = \sum_{k=0}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} ,$$

we can write Eq. (24) as

$$\sum_{i=1}^{n} \omega_i\, D_{n-1}(x_j,x_i) = 1 . \tag{25}$$

To proceed further we must establish the Christoffel-Darboux identity,

$$\frac{Q_n(x)\, Q_{n-1}(y) - Q_{n-1}(x)\, Q_n(y)}{N_{n-1}(x - y)}$$

$$= \frac{[(x - \mu_n)Q_{n-1}(x) - \sigma^2_{n-1}\, Q_{n-2}(x)]Q_{n-1}(y) - Q_{n-1}(x)[(y - \mu_n)Q_{n-1}(y) - \sigma^2_{n-1}Q_{n-2}(y)]}{N_{n-1}(x - y)}$$

$$= \frac{(x - y)\, Q_{n-1}(x)\, Q_{n-1}(y) + \sigma^2_{n-1}\,[Q_{n-1}(x)\, Q_{n-2}(y) - Q_{n-2}(x)\, Q_{n-1}(y)]}{N_{n-1}(x - y)}$$

$$= \frac{Q_{n-1}(x)\, Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-1}(x)\, Q_{n-2}(y) - Q_{n-2}(x)\, Q_{n-1}(y)}{N_{n-1}(x - y)} \cdot \frac{N_{n-1}}{N_{n-2}}$$

$$= \frac{Q_{n-1}(x)\, Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-1}(x)\, Q_{n-2}(y) - Q_{n-2}(x)\, Q_{n-1}(y)}{N_{n-2}(x - y)}$$

$$= \frac{Q_{n-1}(x)\, Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-2}(x)\, Q_{n-2}(y)}{N_{n-2}} + \frac{Q_{n-2}(x)\, Q_{n-3}(y) - Q_{n-3}(x)\, Q_{n-2}(y)}{N_{n-3}(x - y)}$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{Q_1(x)\, Q_0(y) - Q_0(x)\, Q_1(y)}{N_0(x - y)}$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{(x - \mu_1) - (y - \mu_1)}{N_0(x - y)} \qquad (26)$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{1}{N_0} = \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{Q_0(x)\, Q_0(y)}{N_0}$$

$$= \sum_{k=0}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} = D_{n-1}(x,y) \quad .$$

Therefore

$$D_{n-1}(x_j,x_i) = \frac{Q_n(x_j)\, Q_{n-1}(x_i) - Q_{n-1}(x_j)\, Q_n(x_i)}{N_{n-1}(x_j - x_i)} \qquad (27)$$

For $i \neq j$ and $Q_n(x_j) = Q_n(x_i) = 0$,

$$D_{n-1}(x_j,x_i) = 0 \quad .$$

Therefore, returning to Eq. (25),

$$\sum_{i=1}^{n} \omega_i\, D_{n-1}(x_j,x_i) = \omega_j\, D_{n-1}(x_j,x_j) = 1$$

or

$$\omega_j = \{D_{n-1}(x_j,x_j)\}^{-1} = \left[\sum_{k=0}^{n-1} \frac{Q_k^2(x_j)}{N_k}\right]^{-1} \qquad (28)$$

Limits of μ_i and σ_i^2

In the calculations leading to the generalized Gaussian quadrature we obtained two restrictions which had to be satisfied in order to have a positive distribution located on the interval (-1,+1). These restrictions were:

1) $N_i > 0$.

2) All the roots of $Q_i(x)$ lie in the interval (-1,+1).

Let us determine first what limitations these two restrictions place on the quantities μ_i, σ_i^2. Consider first the effect of adding an infinitesimal amount $\Delta\mu$ to μ_i. We have

$$Q_i(x) = (x - \mu_i)\, Q_{i-1}(x) - \sigma_{i-1}^2\, Q_{i-2}(x)$$

and

$$Q_i^*(x) = (x - \mu_i - \Delta\mu)\, Q_{i-1}(x) - \sigma_{i-1}^2\, Q_{i-2}(x) = Q_i(x) - \Delta\mu\, Q_{i-1}(x).$$

If Q_i has a root at x_0, then Q_i^* will have a root at $x_0 + \Delta x_0$

$$Q_i^*(x_0 + \Delta x_0) = 0 = Q_i(x_0 + \Delta x_0) - \Delta\mu\, Q_{i-1}(x_0 + \Delta x_0)\ .$$

If we expand the right-hand side and keep only first order terms

$$0 = Q_i(x_0) + \Delta x_0\, Q_i'(x_0) - \Delta\mu\, Q_{i-1}(x_0) = \Delta x_0\, Q_i'(x_0) - \Delta\mu\, Q_{i-1}(x_0)$$

or

$$\Delta x_0 = \frac{Q_{i-1}(x_0)}{Q_i'(x_0)}\,\Delta\mu \qquad . \tag{29}$$

Since $Q_i(x)$ is positive as x approaches $+\infty$, then $Q_i'(x_0) > 0$ at x_0 equal to the largest root of Q_i. At successively smaller roots of Q_i the sign of $Q_i'(x)$ alternates from positive to negative. $Q_{i-1}(x)$ is similarly positive at $+\infty$. Also, it has no roots greater than the largest root of Q_i. Therefore $Q_{i-1}(x) > 0$ at the largest root of Q_i. Because the roots of Q_{i-1} "interleave" with the roots of Q_i, the sign of $Q_{i-1}(x)$ must

alternate at successive roots of $Q_i(x)$. Therefore, at all roots of $Q_i(x)$ we must have:

$$\frac{Q_{i-1}(x)}{Q_i'(x)} > 0 \qquad (30)$$

or, going back to Equation (29)

$$\frac{dx_0}{d\mu_i} > 0 .$$

Therefore, as μ_i is increased, the roots of $Q_i(x)$ shift to the right, and, as μ_i is decreased, the roots shift downward. If μ_i is steadily increased, the largest root of Q_i will eventually equal 1. This point is determined by

$$Q_i(1) = 0 = (1 - \mu_i)\, Q_{i-1}(1) - \sigma_{i-1}^2\, Q_{i-2}(1)$$

or

$$\mu_i = 1 - \sigma_{i-1}^2 \frac{Q_{i-2}(1)}{Q_{i-1}(1)} .$$

This is clearly the maximum value of μ_i, which will generate positivity in the interval (-1,+1). Likewise there is a minimum value at which the lowest root of Q_i occurs at x = -1.

$$Q_i(-1) = 0 = (-1 - \mu_i)\, Q_{i-1}(-1) - \sigma_{i-1}^2\, Q_{i-2}(-1)$$

or

$$\mu_i^{\min} = -1 - \sigma_{i-1}^2 \frac{Q_{i-2}(-1)}{Q_{i-1}(-1)} .$$

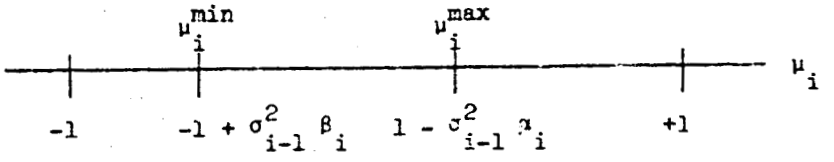
Note that

$$\alpha_i = \frac{Q_{i-2}(1)}{Q_{i-1}(1)} > 0 ,$$

due to the positivity of the functions as they approach $+\infty$ and that

$$\beta_i = -\frac{Q_{i-2}(-1)}{Q_{i-1}(-1)} > 0 ,$$

due to their alternation in sign at $-\infty$. Since $\sigma_{i-1}^2 > 0$, we have the following picture on a μ_i-axis



Now that we have upper and lower limits for μ_i, what can we say about σ_i^2? Since $\sigma_i^2 = N_i/N_{i-1}$, restriction I implies that $\sigma_i^2 > 0$. We can obtain an upper limit to σ_i^2 by setting $\mu_{i+1}^{min} = \mu_{i+1}^{max}$. For larger values of σ_i^2, $\mu_{i+1}^{min} > \mu_{i+1}^{max}$, which means that there is no value of μ_{i+1} which will allow all the roots of $Q_{i+1}(x)$ to lie inside $(-1,+1)$. Thus

$$1 - (\sigma_i^2)_{max} \frac{Q_{i-1}(+1)}{Q_i(+1)} = -1 - (\sigma_i^2)_{max} \frac{Q_{i-1}(-1)}{Q_i(-1)}$$

$$2 = (\sigma_i^2)_{max} \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

$$(\sigma_i^2)_{max} = 2 \Big/ \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

We can work back from the limits on μ_i and σ_i^2 to obtain limits on the moments.

$$\sigma_i^2 = N_i/N_{i-1}$$

$$N_i = \sum_{k=0}^{i} a_{ik} M_{k+i} = M_{2i} + \sum_{k=0}^{i-1} a_{ik} M_{k+i} \quad \text{since } a_{ii} = 1 \ .$$

Therefore

$$0 < \sigma_i^2 < 2 \Big/ \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

implies

$$-\sum_{k=0}^{i-1} a_{ik} M_{k+i} < M_{2i} < \frac{2N_{i-1}}{\left[\frac{Q_{i-1}^{(+1)}}{Q_i^{(+1)}} - \frac{Q_{i-1}^{(-1)}}{Q_i^{(-1)}}\right]} - \sum_{k=0}^{i-1} a_{ik} M_{k+i}$$

$$\mu_{i+1} = \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}}$$

$$L_{i+1} = \sum_{k=0}^{i} a_{ik} M_{k+i+1} = M_{2i+1} + \sum_{k=0}^{i-1} a_{ik} M_{k+i+1}$$

$$\mu_{i+1}^{max} = 1 - c_i^2 \frac{Q_{i-1}^{(1)}}{Q_i^{(1)}} \text{ ; therefore,}$$

$$L_{i+1}^{max} = N_i - N_i c_i^2 \frac{Q_{i-1}^{(1)}}{Q_i^{(1)}} + \frac{N_i L_i}{N_{i-1}} = N_i \left(1 - c_i^2 \frac{Q_{i-1}^{(1)}}{Q_i^{(1)}}\right) + L_i c_i^2$$

$$M_{2i+1} < N_i \left(1 - c_i^2 \frac{Q_{i-1}^{(1)}}{Q_i^{(1)}}\right) + L_i c_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k+i+1}$$

also

$$M_{2i+1} > N_i \left(-1 - c_i^2 \frac{Q_{i-1}^{(-1)}}{Q_i^{(-1)}}\right) + L_i c_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k-i+1} \quad .$$

To obtain the limits on the Legendre coefficients, take the set of moments already determined $M_1, M_2, \ldots, M_{2i-1}$ combined with M_{2i}^{max} and convert from moments to Legendre coefficients. This gives f_{2i}^{max}. When $M_1, M_2, \ldots, M_{2i-1}$ are combined with M_{2i}^{min} and converted, one obtains f_{2i}^{min}.

4.12. REFERENCES

1. K. D. Lathrop, "DTF-IV, A FORTRAN-IV Program for Solving the Multigroup Transport Equation with Anisotropic Scattering," LA-3373, Los Alamos Scientific Laboratory (1965).

2. W. W. Engle, Jr., "A User's Manual for ANISN," K-1693 (1967).

3. W. A. Rhoades and F. R. Mynatt, "The DOT-III Two-Dimensional Discrete Ordinates Transport Code," ORNL-TM-4280 (1973).

4. W. Guber, et al., "A Geometric Description Technique Suitable for Computer Analysis of Both the Nuclear and Conventional Vulnerability of Armored Military Vehicles," MAGI-6701 (August, 1967).

5. M. O. Cohen, W. Guber, et al., "SAM-CE - A Three Dimensional Monte Carlo Code for the Solution of the Forward Neutron and Forward and Adjoint Gamma Ray Transport Equations," MR-7021 (DNA2830F) pp. 3.3-3.18 (November, 1971).

6. D. C. Irving, R. M. Freestone, Jr., and F. B. K. Kam, "Ø5R, A General Purpose Monte Carlo Neutron Transport Code," ORNL-3622 (1965).

7. C. L. Thompson and E. A. Straker, "Ø6R-ACTIFK, Monte Carlo Neutron Transport Code," ORNL-CF-69-8-36 (1969).

8. D. C. Irving, "Description of the CDC-1604 Version of the Ø6R Neutron Monte Carlo Transport Code," ORNL-CF-71-5-14 (1971).

9. C. E. Burgart and E. A. Straker, "XCHEKR - A Multigroup Cross Section Editing and Checking Code," ORNL-TM-3518 (1971).

10. F. H. Clark and N. A. Betz, "Importance Sampling Devices for Selecting Track Lengths and Directions After Scatter in Ø5R," ORNL-TM-1484 (1966).

11. F. H. Clark, "The Exponential Transform as an Importance-Sampling Device - A Review," ORNL-RSIC-14 (1966).

12. V. R. Cain, E. A. Straker, and G. Thayer, "Monte Carlo Path Length Selection Routines Based on Some Specific Forms of the Importance Function," ORNL-TM-1967 (1969).

13. M. B. Emmett, C. E. Burgart and T. J. Hoffman, "DOMINO, A General Purpose Code for Coupling Discrete Ordinates and Monte Carlo Radiation Transport Calculations," ORNL-4853 (1973).

14. Private Communication from W. W. Engle, Jr., ORNL.

15. F. H. Clark, Nucl. Sci. Eng. 27, 235-239 (1967).

16. T. J. Tyrrell, "TDUMP - A Translation Routine for Use in Dumping FORTRAN Arrays," ORNL-CF-70-7-8 (1970).

PART V

Program PICTURE

TABLE OF CONTENTS

LIST OF TABLES

5.1 ABSTRACT

The PICTURE program was written to provide aid in preparing correct input data for the combinatorial geometry package CG. It provides a printed view of arbitrary two-dimensional slices through the geometry. By inspecting these pictures one may determine if the geometry specified by the input cards is indeed the desired geometry. This report describes PICTURE, its options and input.

5.2 Introduction

The PICTURE program was devised to help the user determine if his geometry input data does indeed describe the geometry he had in mind. PICTURE displays, as printed output, two-dimensional slices through the specified geometry. A regularly spaced array of points is generated and each point is plotted as a symbol related to either media, region or zone depending on the option selected by the user. By printing out this array a rough picture of one view of the geometry is produced. The user may then look at the picture and determine if the geometry is as intended. A sample problem is contained in Part 3, A Sample Problem Notebook.

5.3 ROUTINES

5.3.1 Main Program or PICTURE

The executive routine for the PICTURE program reads in the input, calculates the coordinates of the picture to be plotted and controls the calls to other routines. There are several different ways in which a two-dimensional slice through the geometry may be obtained. These different options are discussed in Section 5.5 with discussions of the input. The characters to be printed for corresponding media or regions may be changed by altering the values in ATABLE as given in the data statement.

Subroutines called: JØMIN, PRINT
Commons required: PICT
Variables required: Several input cards are read.
Variables changed: All variables in common PICT. (see Table 5.1)
Significant internal variables:

- INT - input logical unit,
- IØT - output logical unit,
- NADD(1) - first location in blank common for storage of geometry data,
- NCK - flag to indicate which set of input options was used to define the two-dimensional slice.

Table 5.1. Definition of Variables in Common PICT

Variable	Definition
DELU	The increment in geometry units between lines in the picture in the U direction.
DELV	The increment in geometry units between lines in the picture in the V direction.
XO, YO, ZO	The coordinates in geometry units defining the first point (upper left hand corner) of the picture.
XU, YU, ZU	The length (in direction U) of the picture in geometry units.
XV, YV, ZV	The width (in direction V) of the picture in geometry units.
NSTØR(130)	An array used to store the medium, region, or zone number for one line of the picture.
IRG	A flag indicating that region, zone, or medium geometry parameter should be printed if IRG is negative, zero, or positive, respectively.

5.3.2 Subroutines

Subroutine PRINT (KXX, KYY, ATABLE) -

This routine controls the printing of the picture. First it is determined if the picture is to be more than one "page" (130 columns) wide and then for each line of the picture on the first page the information to be printed is determined by calling MESH and printed with a 130A1 format. The next lines are then calculated and printed until that "page" is finished. Note that a page refers to width, not length. Thus, as much detail may be obtained as necessary in both directions by piecing together the output.

Called from: PICTURE (Main)
Subroutine called: MESH
Commons required: PICT
Variables required:

- KXX - number of intervals in the U direction (direction of paper movement through the printer),
- KYY - number of intervals in the V direction (line),
- ATABLE - table of characters to be printed.

Significant interval variable:

- NPAGES - number of subpictures required to cover the width of the total picture,
- IØT - output logical unit,
- NV - number of characters per line (characters/page width).

Subroutine MESH (XS, YS, ZS, NV) -

Subroutine MESH is used by the PICTURE package to set up one line of print in the array NSTØR. Both for efficiency and to debug the combinatorial geometry package†, this version has been modified to work exactly like particle tracking. LØØKZ is first called to determine the zone of the first grid point. A trajectory to the last grid point is then initialized, and successive calls to G1 "track" a particle to the last point, setting the region of each grid point in NSTØR. By setting IRG negative, zero, or positive, either NREG, IR, or NMED will be stored in the print array NSTØR.

Called from: PRINT

Subroutines called: LØØKZ, G1

Commons required: PICT, GØMLØC, PAREM, ØRGI, blank.

Variables required:

XS, YS, ZS - coordinates of first grid point,
NV - number of grid points,
DELV - distance between grid points,
IRG - flag to print NREG, IR, or NMED if IRG is negative, zero, or positive.

Variables changed: NSTØR - print array.

Significant internal variables:

J - grid point index,
ISTØR - value to be stored in NSTOR between successive boundary crossings.

†See Part 4, Section 4.7 of this document.

5.4 Input Data

1. Card PA: Format (I5)

   NUSE: The number of characters to be read on Card B to replace the standard values of ATABLE. Leave Card A blank and omit Card B if the standard ATABLE is desired. NUSE ≤ 50.

2. Card PB: Format (50A1) (omit if NUSE = 0)

   ATABLE(I), I=1, NUSE: The list of characters that are to be printed for each medium. For medium N, ATABLE (N+1) is printed. If N ≥ 47, ATABLE (48) is printed. The standard values of ATABLE are:

Medium Number	Character Printed
0 (external void)	
1 through 9	1 through 9
10 through 35	A through Z
36 through 46	various special characters
≥ 47 (including internal voids)	(blank)

3. GEØM input: Combinatorial geometry input.†

4. Card PC: Format (2I2, 18A4)

INCT:
- = 0 After this picture, return to Card PC for another picture with the same geometry.
- = 1 After this picture, read in a new GEØM input. (step 3 above)

IRG:
- = -1 Display the region geometry.
- = 0 Display the zone geometry.
- = 1 Display the material geometry.

TITLE(I), I=1, 18: 72 characters to be printed as a title.

---

†See description of geometry input in Section 4.3 of Part 4.

5. Card PD: Format (6E10.5)

X_{UL}, Y_{UL}, Z_{UL} — X, Y, and Z coordinates in the combinatorial geometry of the upper left corner of the picture.

X_{LR}, Y_{LR}, Z_{LR} — X, Y, and Z coordinates in the combinatorial geometry of the lower right corner of the picture.

Note: Card PD partially describes the plane of the slice by defining two points in the plane and designates the top, bottom, left and right sides of the picture.

6. Card PE: Format (6E10.5)

U_X, U_Y, U_Z — Direction numbers proportional to the direction cosines for the U axis of the picture. The U axis points down the printed page in the direction the page moves through the printer.

V_X, V_Y, V_Z — Direction numbers for the V axis of the picture. The V axis points to the right across the page.

NOTE: Card PE completes the description of the plane of the slice by giving a line in the plane, also specifies the orientation of the picture on the output.

7. Card PF: Format (2I5, 2E10.5)

NU: Number of intervals to print along the U axis (overrides DELU).

NV: Number of intervals to print along the V axis (overrides DELV).

DELU: Spacing (in GEØM units) of intervals along the U axis.

DELV: Spacing (in GEØM units) of intervals along the V axis.

NOTE: All four entries are not required as input on Card PF; see below for explanation.

5.5 Options

1. $X_{LR} = Y_{LR} = Z_{LR} = 0.$

For this case NU and NV must be specified. In addition, either DELU or DELV must be specified. If the other is left blank, the code will produce an undistorted picture. If both DELU and DELV are specified the picture is likely to be distorted. The standard printers give 10 characters to the inch across a line but only 6 lines per inch down the page. Because of this DELV = .6*DELU is necessary to produce an undistorted picture.

2. X_{LR} or Y_{LR} or $Z_{LR} \neq 0.$

If any one variable on Card PD is specified, the code will calculate the others to produce an undistorted picture.

If both NU and DELU (or both NV and DELV) are specified, DELU (or DELV) will be ignored.

The U and V axes may have arbitrary orientation. (If they are not orthogonal, the resulting picture will be distorted.) In Option 1, the first point will be at $(X, Y, Z)_{UL}$, and the remaining points in the directions and at the distances specified. In Option 2, the range from X_{UL} to X_{LR} is divided into intervals and the calculated points will be at the midpoints of the intervals. The first point will be 1/2 interval past $(X, Y, Z)_{UL}$ and the final point will be within 1/2 interval of $(X, Y, Z)_{LR}$. If $(X, Y, Z)_{LR}$ does not lie on the U-V plane, or if the U and V axes are not orthogonal, the location of the final point is not readily predictable.

The simplest method to obtain the correct results is to specify two diagonal corners of the plane of the slice on Card PD, with the top having the short dimension. Then, on Card PE, specify the U axis to be parallel to the edge of the slice with the large dimension (left or right side), and the V axis to be parallel to the edge of the slice with the small dimension (top or bottom). Finally, let the only entry on Card PF be NV equal to maximum number of characters per line on your printer; this will provide the largest undistorted picture.