221
10-6

DWR 1572

ORNL-4585
UC-80 – Reactor Technology

THE MORSE CODE – A MULTIGROUP NEUTRON AND GAMMA-RAY MONTE CARLO TRANSPORT CODE

E. A. Straker
P. N. Stevens
D. C. Irving
V. R. Cain

OAK RIDGE NATIONAL LABORATORY
operated by
UNION CARBIDE CORPORATION
for the
U.S. ATOMIC ENERGY COMMISSION

BLANK PAGE

Printed in the United States of America. Available from Clearinghouse for Federal
Scientific and Technical Information, National Bureau of Standards,
U.S. Department of Commerce, Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.65

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

BLANK PAGE

ORNL-4585

Neutron Physics Division

THE MORSE CODE - A MULTIGROUP NEUTRON AND GAMMA-RAY MONTE CARLO TRANSPORT CODE

E. A. Straker, P. N. Stevens,*
D. C. Irving,† and V. R. Cain

*University of Tennessee, Knoxville, Tennessee.

†Present address: Savannah River Laboratory, Aiken, South Carolina 29802

NOTE:
This Work Partially Funded by
DEFENSE ATOMIC SUPPORT AGENCY
Under Subtask PE08001

SEPTEMBER 1970

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

TABLE OF CONTENTS

PAGE

PAGE

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, as well as specialized one-dimensional geometry descriptions, 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 in the appendices. Logical flow charts for each subroutine add to the understanding of the code.

LIST OF TABLES

I. Introduction

The Multigroup Oak Ridge Stochastic Experiment code (MØRSE) 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, one-, two-, and three-dimensional geometry packages, 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. (A set of multigroup neutron cross sections - 99 group, P_8 - based on ENDF/B is available from the Radiation Shielding Information Center;[2] likewise, some coupled neutron gamma-ray sets are also available from RSIC.)

Cross sections may be read in either the DTF-IV[3] format or ANISN[4] and DOT[5] format. The auxiliary information giving the number of groups, elements

*A versatile analysis package, SAMBO, which handles most of the drudgery associated with estimation from random walk events is described in ref. 1.

BLANK PAGE

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 Appendix A and are written as Equations (40) and (95).

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

Input to MØRSE is read in five separate modules: (1) walk; (2) cross section; (3) user; (4) source; and (5) geometry. The walk input is read in subroutine INPUT 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. Input information required for analysis of the histories must be read by a user-written subroutine SCØRIN which is called from BANKR. Since the source varies from problem to problem, input may also be read in a user-written subroutine SØRIN for the definition of the source. The geometry input is read by subroutine JØMIN.

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 the user-written routine NRUN.

Figure 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. 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 any of the geometry packages written for Ø5R,[6,7] including the general three-dimensional geometry. Slight modifications have been made to include variable input-output units and to include the logic for albedo scattering.

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

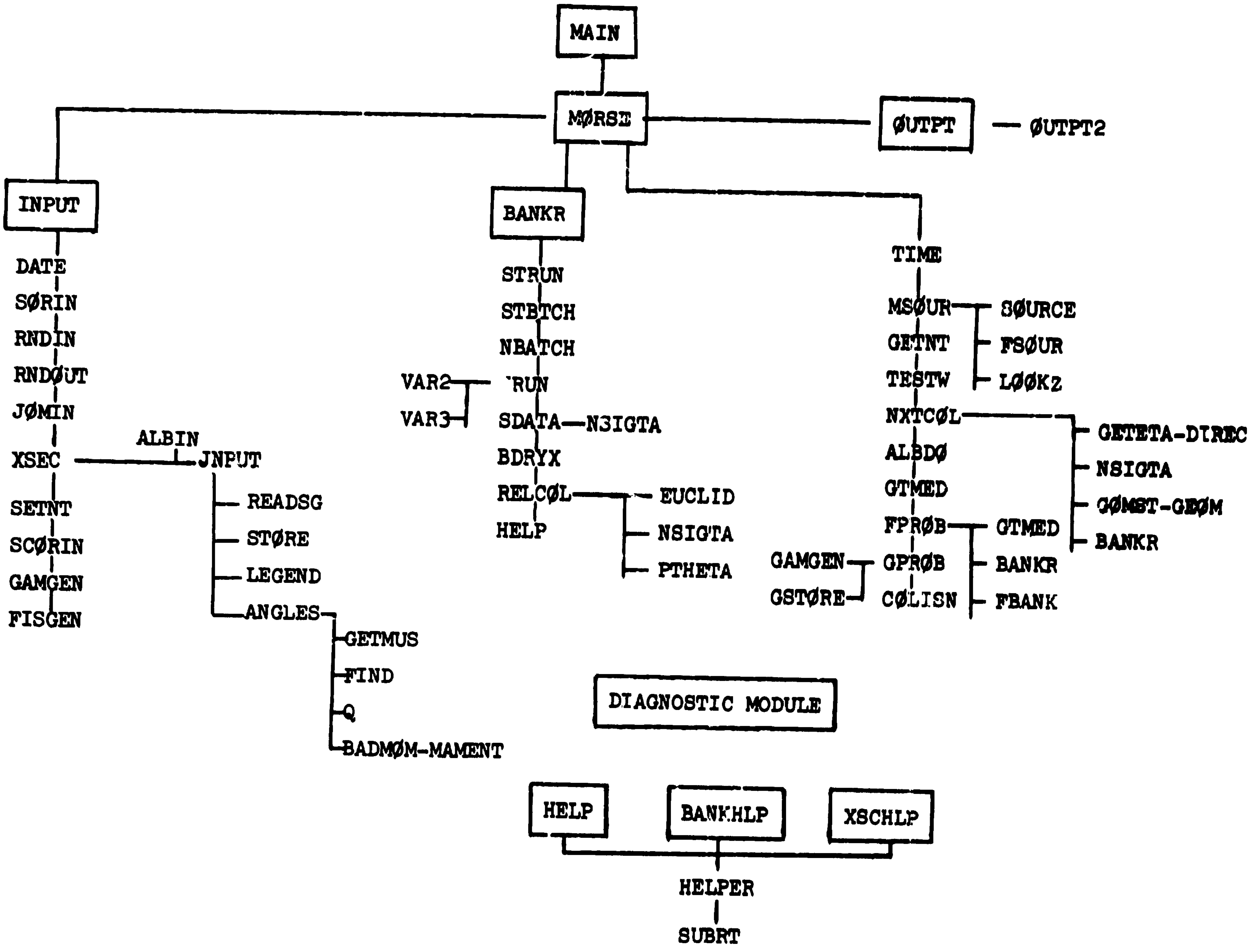


Fig. 1. Hierarchy of Subroutines in the MØRSE Code.

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 Ø5R 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.

Detailed descriptions of the subroutines with the logical flow charts are found on the following pages. The appendices contain detailed derivations of various forms of the transport equation, a detailed derivation of the treatment of the angular distribution of scattering, and a detailed description of the required input.

II. Random Walk Module

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 cnly two main labelled commons (APØLLØ and NUTRØN) in the random walk routines. Tables I and II list the definitions of the variables in these two commons. Note that in Table II "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. 2, along with definitions in Table III. The locations of the variables are given in Table IV. All the variables used as location labels, except NGEØM, locate cell zero of an array. Cells NLAST + 1 to NLAST + NLEFT are available to the user for analysis arrays.

A description and a logical flow chart for the subroutines that make up the random walk module are given the following pages in this chapter.

Table I. 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
UIMP, VIMP, WIMP	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)	Not used
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
ISØUR	Input source energy group if > 0; otherwise, SØRIN is called to read input spectrum
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. 2 and Table IV)
LØCFWL	Starting location in blank common of the fission weights
LØCEPR	Starting location in blank common of the energy-biasing parameters

Table I (cont.)

Variable	Definition
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
MAXTIM	The elapsed clock time at which the problem is terminated
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
NGPQTG*	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
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

*See page 11 for diagram of energy group structure.

Table I (cont.)

Variable	Definition
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 (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

*See page 11 for diagram of energy group structure.

Table I (cont.)

Variable	Definition
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(10)	Not used.

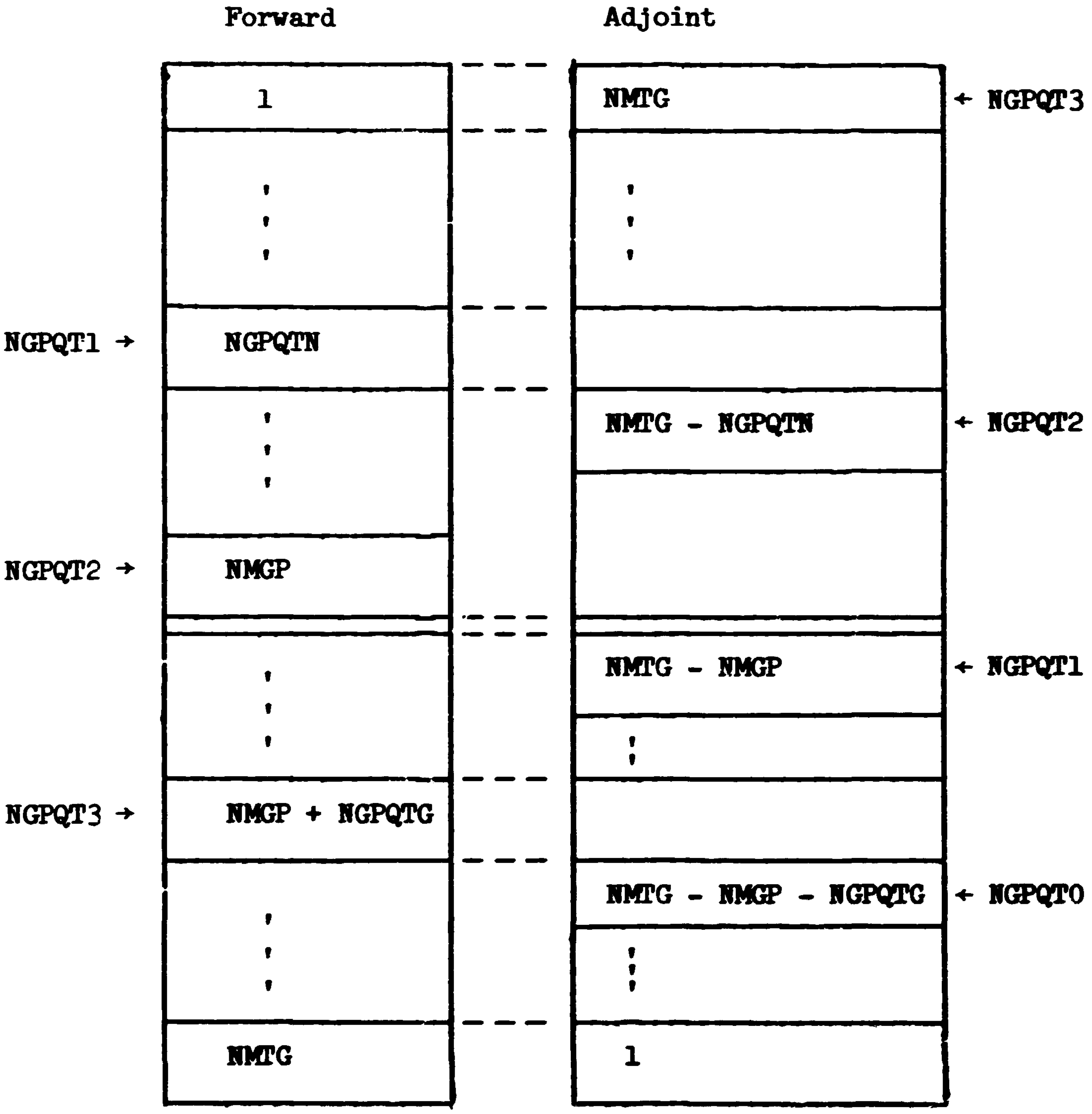


Diagram of Energy Group Structure

Table II. Definition of Variables in NUTRØN 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.
WTBC	Weight just before current collision.
BLZNT	Current block and zone number (packed).
BLZØN	Previous block and zone number (packed).
AGE	Current age.
ØLDAGE	Previous age.

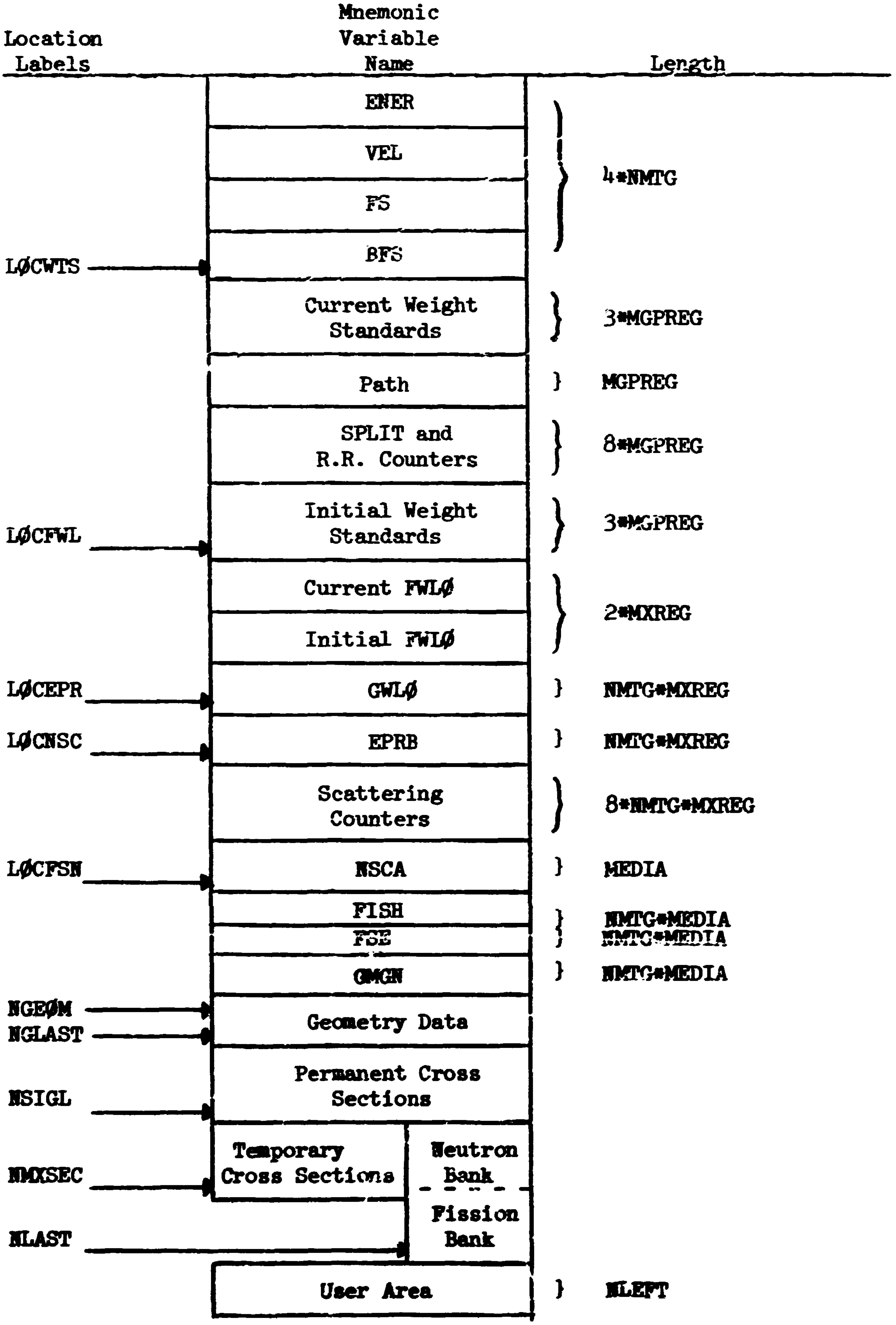


Fig. 2. Layout of Blank Common

Table III. Definitions of Variables in 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.
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.

Table III (cont.)

Mnemonic Variable Name	Definition
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 valuesof 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).
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).
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 IV. Location of 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
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(I); I = LØCNSC + 2*NMTG*MXREG + (NREG-1)*NMTG + IG
WALB(IG,NREG)	BC(I); I = LØCNSC + 3*NMTG*MXREG + (NREG-1)*NMTG + IG

Table IV (cont.)

Mnemonic Variable Name	Location of Array in Blank Common (BC(I) or NC(I))
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

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 (48484848_{16}) in blank common and all labelled commons present in this routine;
4. Sets the two variables used for input and output logical units; and
5. Calls MØRSE for the actual administration of the jcb. (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)

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.

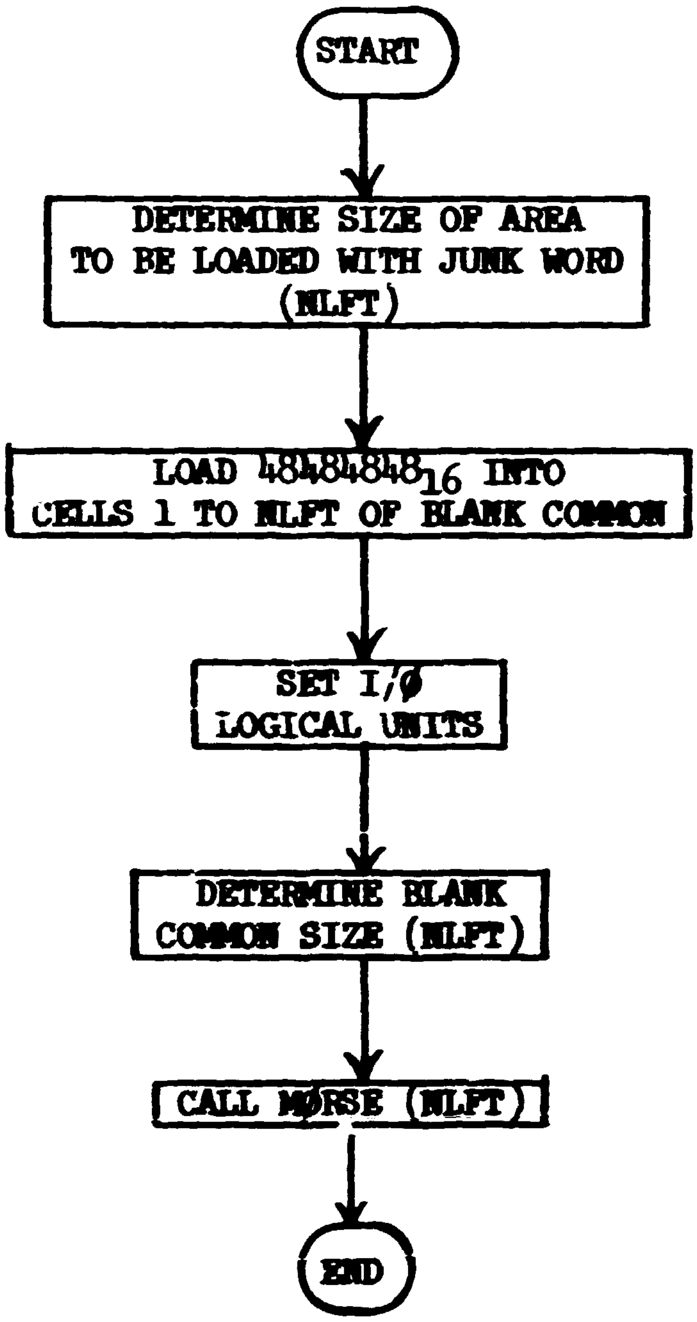
Helpful Hints:

1. Note that if a new cross section, geometry, or analysis package is used, the labelled commons here may have to be modified correspondingly.
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, since the LØC function is used to obtain this value to be transferred to MØRSE.

*See Appendix E for a description of all library routines used in MORSE.

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, and in the desired order, in the common area.
6. The program size, in bytes, is usually on the order of 150000 + 4* (blank common size in words).

Main Routine


START
DETERMINE SIZE OF AREA TO BE LOADED WITH JUNK WORD (NLFT)
LOAD 48484848_{16} INTO CELLS 1 TO NLFT OF BLANK COMMON
SET I/Ø LOGICAL UNITS
DETERMINE BLANK COMMON SIZE (NLFT)
CALL MØRSE (NLFT)
END

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), MSØUR, 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.

Commons required: Blank, APØLLØ, NUTRØN, FISBNK.

Variables required: NLFT, NKILL, NSPLT, NGPQTN, NGPQTG, NITS, NQU1T, NSTRT, NFISH, ITSTR, NMEM, MAXTIM, TCUT } from common APØLLØ (see page 8)

NALB - index indicating that an albedo collision has occurred.

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

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

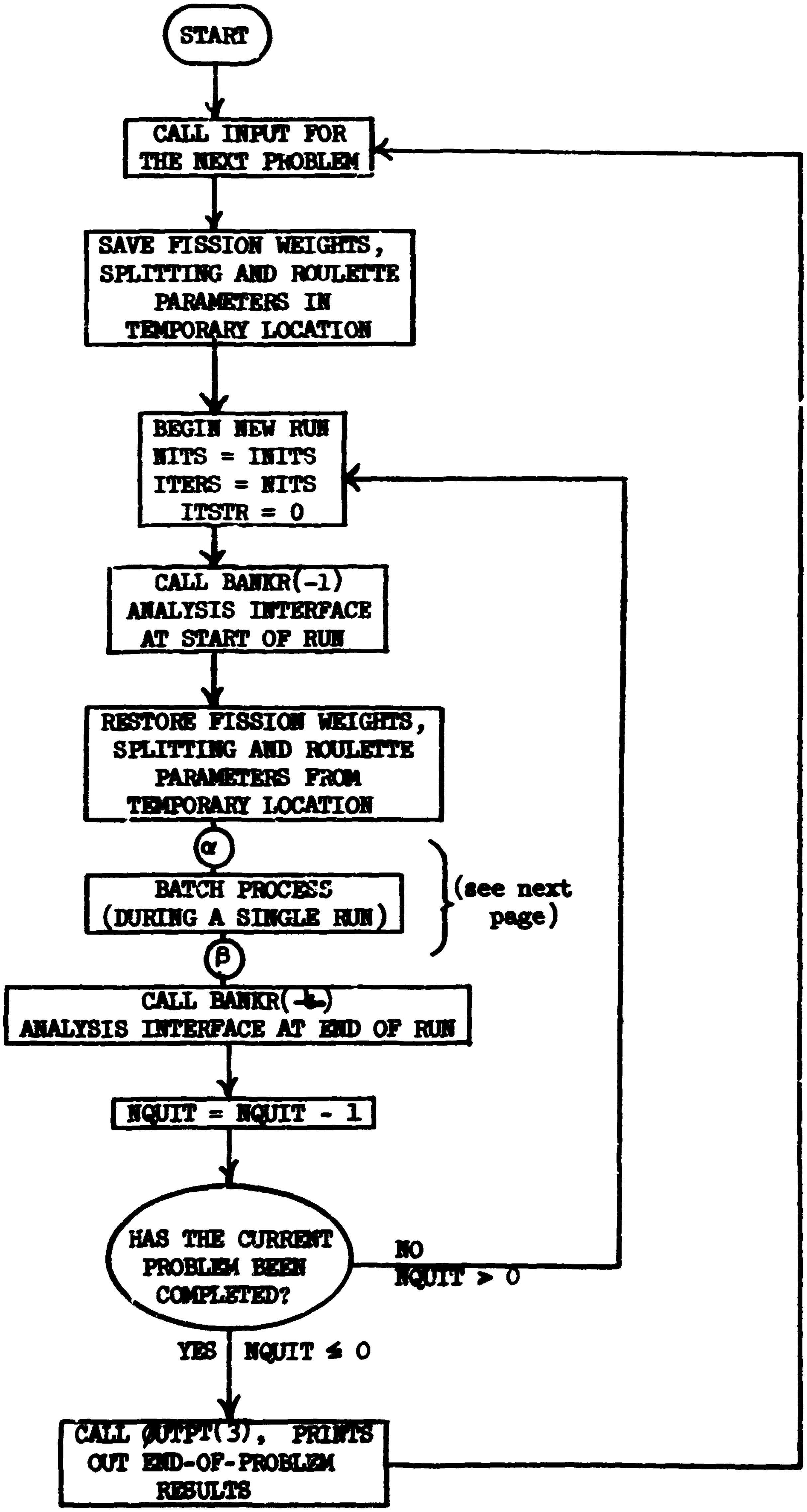
I = 5 - number of real collisions

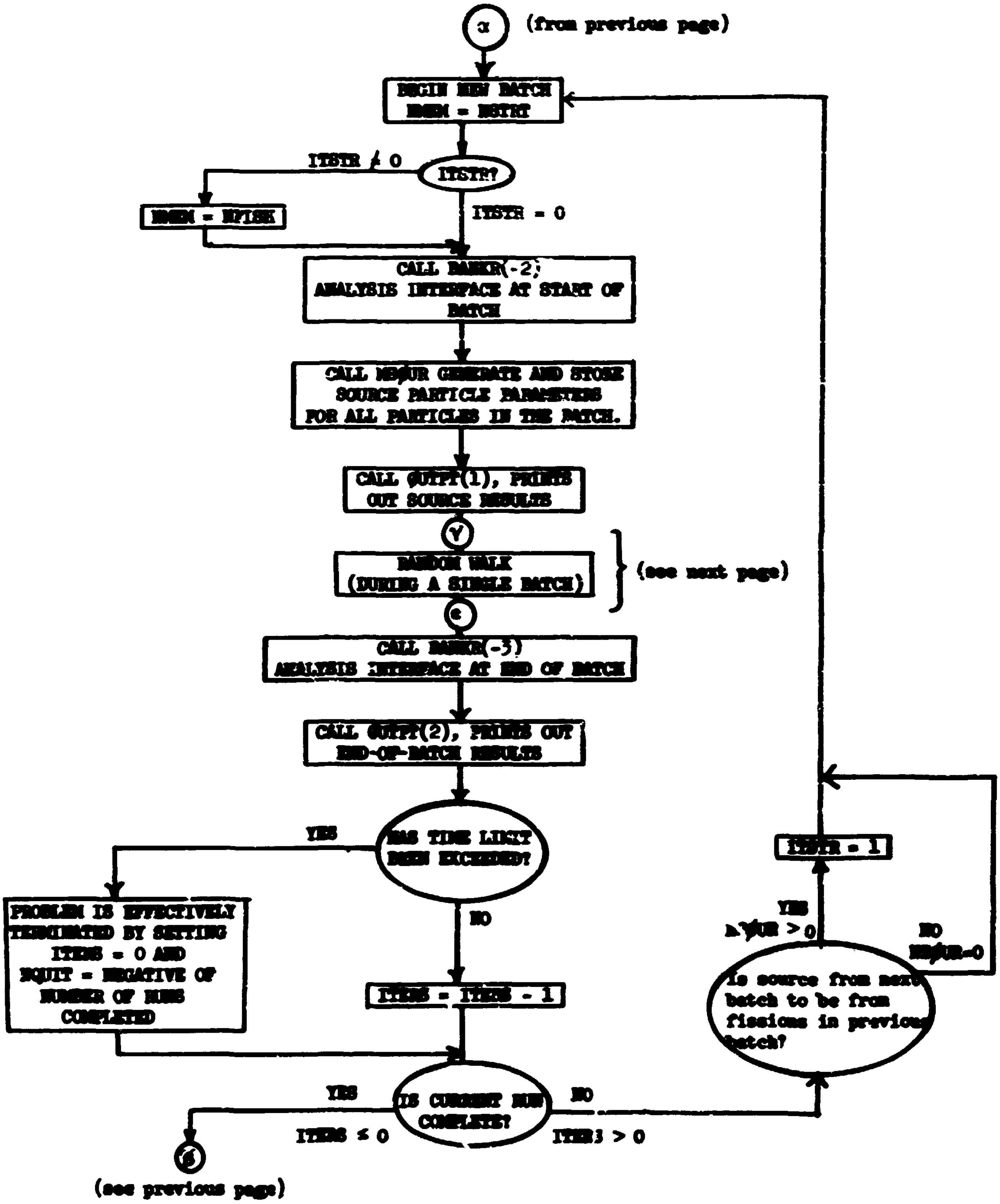
= 6 - number of albedo collisions

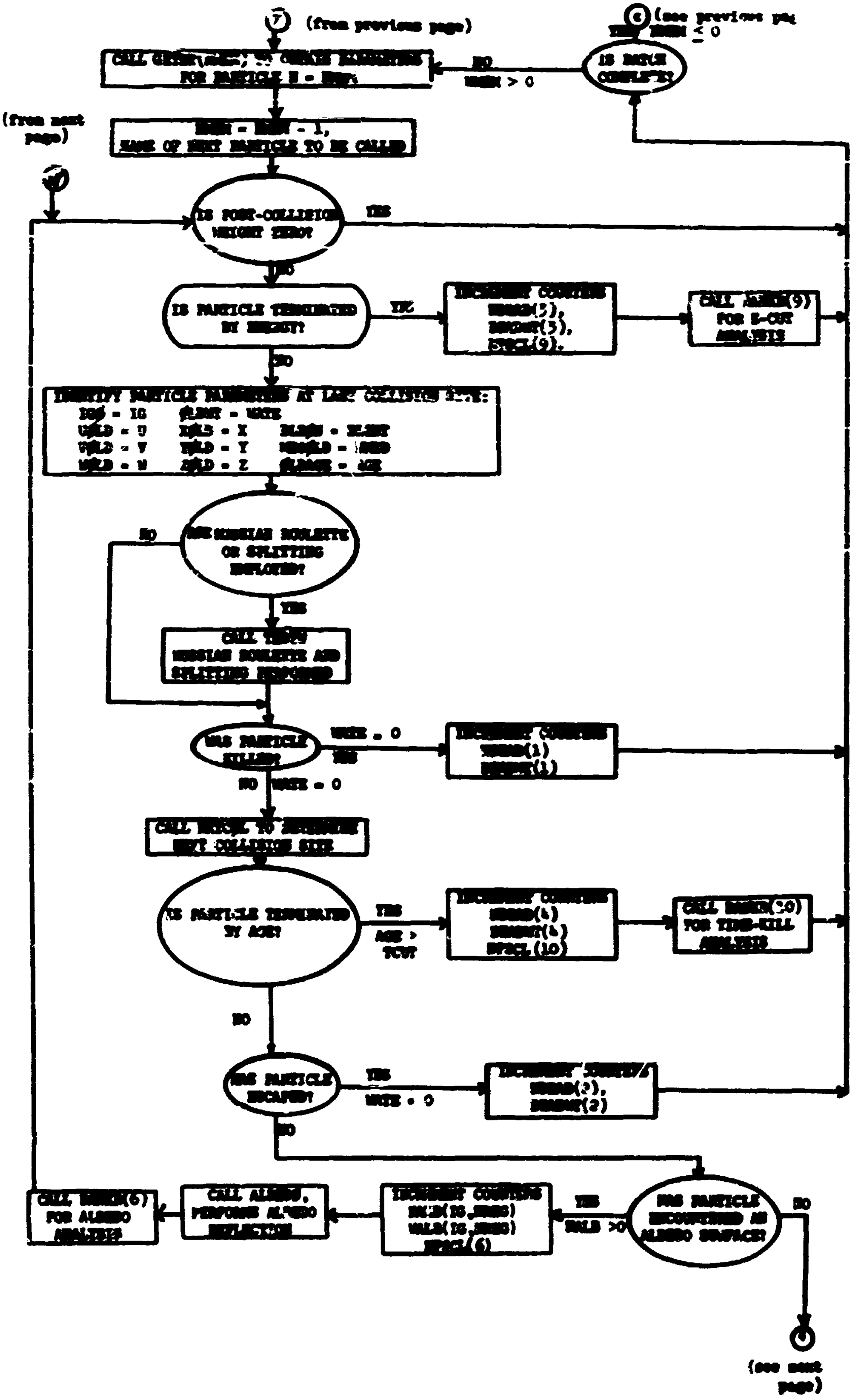
= 9 - number of energy deaths

= 10 - number of age terminations.

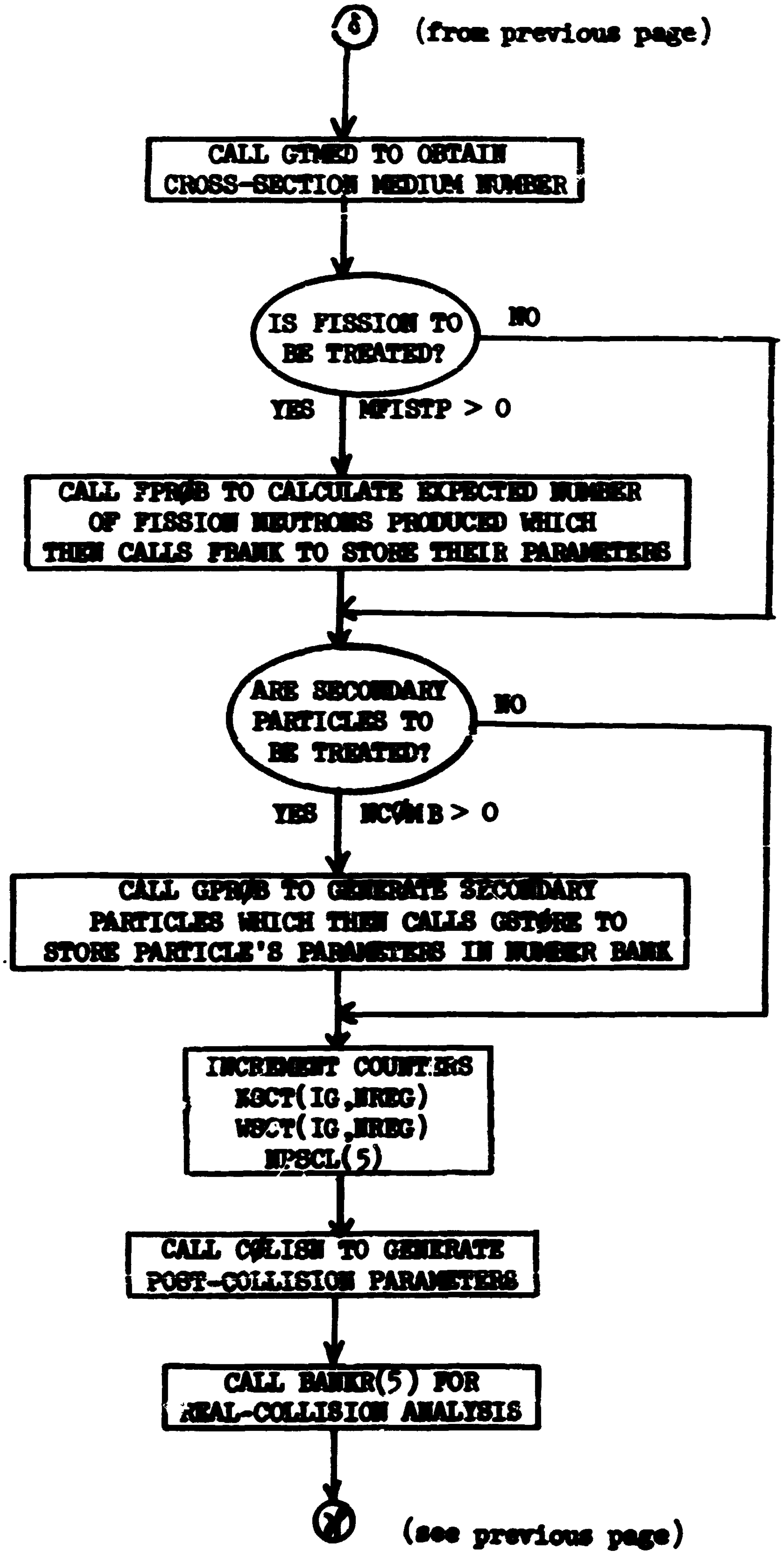
Subroutine MØRSE(NLFT)


START
CALL INPUT FOR THE NEXT PROBLEM
SAVE FISSION WEIGHTS, SPLITTING AND ROULETTE PARAMETERS IN TEMPORARY LOCATION
BEGIN NEW RUN
NITS = INITS
ITERS = NITS
ITSTR = 0
CALL BANKR(-1)
ANALYSIS INTERFACE AT START OF RUN
RESTORE FISSION WEIGHTS, SPLITTING AND ROULETTE PARAMETERS FROM TEMPORARY LOCATION
α
BATCH PROCESS (DURING A SINGLE RUN)
(see next page)
β
CALL BANKR(-4)
ANALYSIS INTERFACE AT END OF RUN
NQUIT = NQUIT - 1
HAS THE CURRENT PROBLEM BEEN COMPLETED?
NO
NQUIT > 0
YES
NQUIT ≤ 0
CALL ØUTPT(3), PRINTS OUT END-OF-PROBLEM RESULTS


α
(from previous page)
BEGIN NEW BATCH
NMEM = NSTRT
ITSTR ≠ 0
ITSTR?
NMEM = NFISH
ITSTR = 0
CALL BANKR(-2);
ANALYSIS INTERFACE AT START OF BATCH
CALL MSØUR GENERATE AND STORE SOURCE PARTICLE PARAMETERS FOR ALL PARTICLES IN THE BATCH.
CALL ØUTPT(1), PRINTS OUT SOURCE RESULTS
γ
RANDOM WALK (DURING A SINGLE BATCH)
(see next page)
ε
CALL BANKR(-3)
ANALYSIS INTERFACE AT END OF BATCH
CALL ØUTPT(2), PRINTS OUT END-OF-BATCH RESULTS
YES
HAS TIME LIMIT BEEN EXCEEDED?
PROBLEM IS EFFECTIVELY TERMINATED BY SETTING ITERS = 0 AND NQUIT = NEGATIVE OF NUMBER OF RUNS COMPLETED
NO
ITERS = ITERS - 1
YES
IS CURRENT RUN COMPLETE?
NO
ITERS ≤ 0
ITERS > 0
δ
(see previous page)
ITSTR = 1
YES
NSØUR > 0
NO
NSØUR=0
Is source from next batch to be from fissions in previous batch?


(from previous page)
(see previous pag
≤ 0
IS BATCH COMPLETE?
NO
> 0
CALL
FOR PARTICLE N =
(from next page)
- 1,
NAME OF NEXT PARTICLE TO BE CALLED
IS POST-COLLISION WEIGHT ZERO?
YES
NO
IS PARTICLE TERMINATED BY ENERGY?
YES
INCREMENT COUNTERS
(3),
(3),
(9).
CALL (9)
FOR E-CUT ANALYSIS
NO
IDENTIFY PARTICLE PARAMETERS AT LAST COLLISION SITE:
= IG
ØLDWT = WATE
UØLD = U
XØLD = X
VØLD = V
YØLD = Y
WØLD = W
ZØLD = Z
ØLDAGE = AGE
NO
ARE RUSSIAN ROULETTE OR SPLITTING EMPLOYED?
YES
CALL
RUSSIAN ROULETTE AND SPLITTING PERFORMED
WAS PARTICLE KILLED?
WATE = 0
YES
INCREMENT COUNTERS
(1)
(1)
NO
WATE = 0
CALL TO DETERMINE
NEXT COLLISION SITE
IS PARTICLE TERMINATED BY AGE?
YES
AGE > TCUT
INCREMENT COUNTERS
(4)
(4)
NPSCL(10)
CALL (10)
FOR TIME-KILL ANALYSIS
NO
WAS PARTICLE ESCAPED?
YES
WATE = 0
INCREMENT COUNTERS
(2),
(2)
NO
HAS PARTICLE ENCOUNTERED AN ALBEDO SURFACE?
YES
WALB > 0
NO
INCREMENT COUNTERS
(IG,NREG)
WALB(IG,NREG)
NPSCL(6)
CALL ALBEDO,
PERFORMS ALBEDO REFLECTION
CALL (6)
FOR ALBEDO ANALYSIS
(see next page)

MØRSE (Random Walk, cont.)


8
(from previous page)
CALL GTMED TO OBTAIN CROSS-SECTION MEDIUM NUMBER
IS FISSION TO BE TREATED?
NO
YES
MFISTP > 0
CALL FPRØB TO CALCULATE EXPECTED NUMBER OF FISSION NEUTRONS PRODUCED WHICH THEN CALLS FBANK TO STORE THEIR PARAMETERS
ARE SECONDARY PARTICLES TO BE TREATED?
NO
YES
NCØMB > 0
CALL GPRØB TO GENERATE SECONDARY PARTICLES WHICH THEN CALLS GSTØRE TO STORE PARTICLE'S PARAMETERS IN NUMBER BANK
INCREMENT COUNTERS
NSCT(IG,NREG)
WSCT(IG,NREG)
NPSCL(5)
CALL CØLISN TO GENERATE POST-COLLISION PARAMETERS
CALL BANKR(5) FOR REAL-COLLISION ANALYSIS
7
(see previous page)

Subroutine DATE (A,NW)

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:

```
      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: INPUT

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

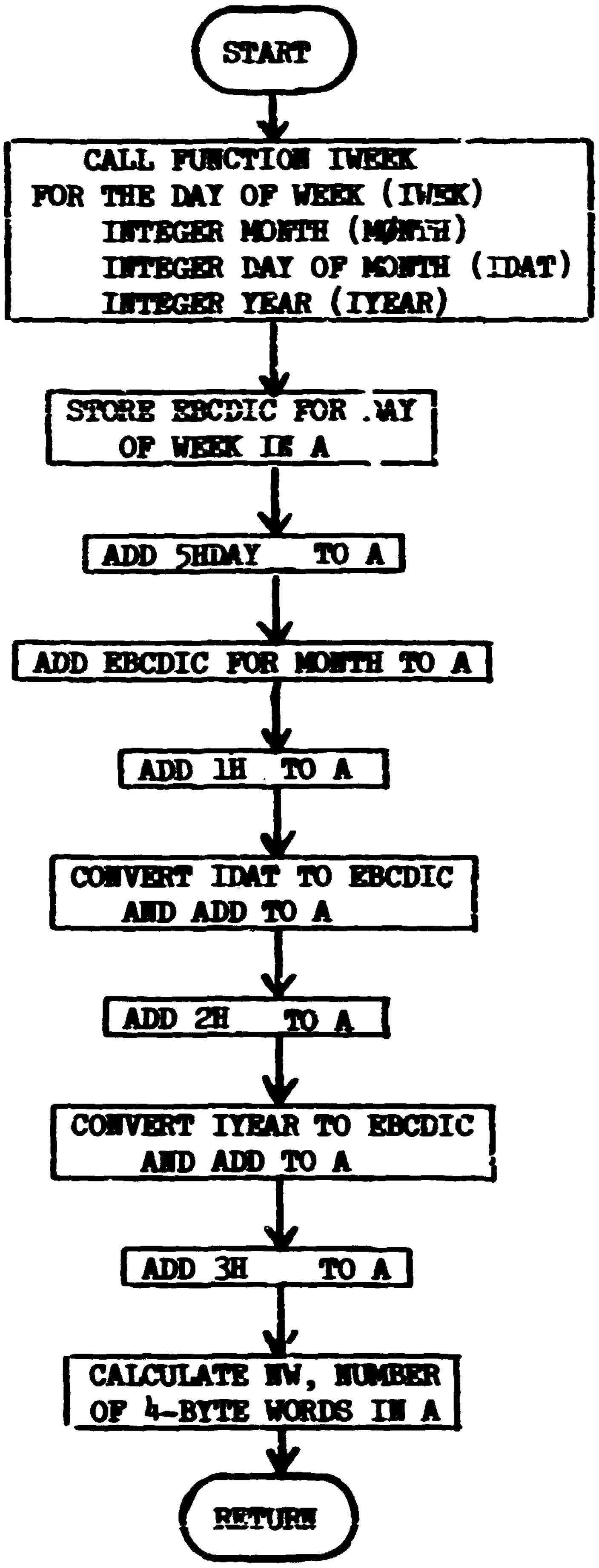
Commons: DATDAT which contains arrays of EBCDIC characters for months and weekdays, arrays of numbers of EBCDIC characters and starting points. It is loaded in a Block Data routine with the following values:

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

	XMØNTH	WEKE	DAY				
Index	(REAL*8)	(REAL*8)	(REAL*8)	IMØNTH	NMØNTH	IWEKE	IWEEK
1	JANUARY①*	HUH?SUN①	DAY,①19①	0	7	0	4
2	FEBRUARY	MØN①TUES		8	8	4	3
3	MARCH③	WEDNES②		16	5	8	3
4	APRIL③	THURS③		24	5	12	4
5	MAY①JUNE	FRI⑤		32	3	16	6
6	JULYAUGU	SATUR③		36	4	24	5
7	ST②SEPT			40	4	32	3
8	EMBER③			44	6	40	5
9	ØCTØBER①			52	9		
10	NØVEMBER			64	7		
11	DECEMBER			72	8		
12				80	8		

*Ⓝ denotes N blanks.

Subroutine DATE (A,NW)


START
CALL FUNCTION IWEEK
FOR THE DAY OF WEEK (IWEK)
INTEGER MONTH (MØNTH)
INTEGER DAY OF MONTH (IDAT)
INTEGER YEAR (IYEAR)
STORE EBCDIC FOR DAY OF WEEK IN A
ADD 5HDAY TO A
ADD EBCDIC FOR MONTH TO A
ADD 1H TO A
CONVERT IDAT TO EBCDIC AND ADD TO A
ADD 2H TO A
CONVERT IYEAR TO EBCDIC AND ADD TO A
ADD 3H TO A
CALCULATE NW, NUMBER OF 4-BYTE WORDS IN A
RETURN

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

This routine is provided for the user to determine the number of mean free paths between two points in the system. It will eitner return the total number of mean free paths or will return the first boundary intersection point and the nuwber of mean free paths to that point.

Called from: GETETA

Subroutines called: GEØM, LØØKZ, NSIGTA.

Functions used: DSQRT (library)

Commons required: GEØMC.

Variables required:

MRK - Set to 1 upon calling,

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.

Variables changed:

MRK = 1 for a flight reaching the end point,

= 0 for a flight crossing a medium boundary (NT ≠ 0 only),

= -1 for a flight escaping the system,

= -2 for a flight encountering an internal void (NT ≠ 0 only),

X1, Y1, Z1 = returns boundary intersection point if NTD ≠ 0,

ARG - negative of number of mean free paths,

NTD - if NTD ≠ 0 on input, will return as -1 if an escape occurs,

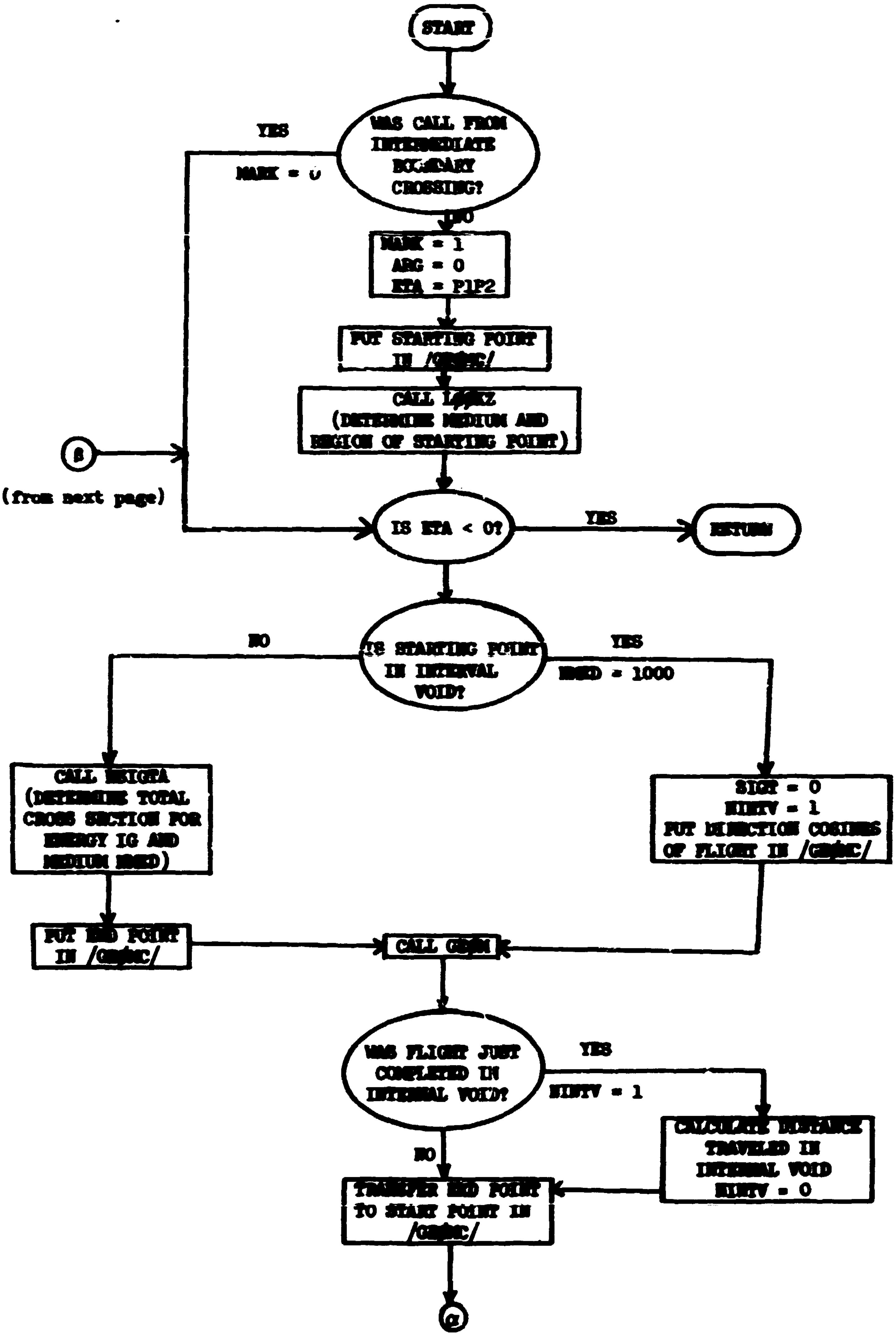
MEDIUM - medium number of end point.

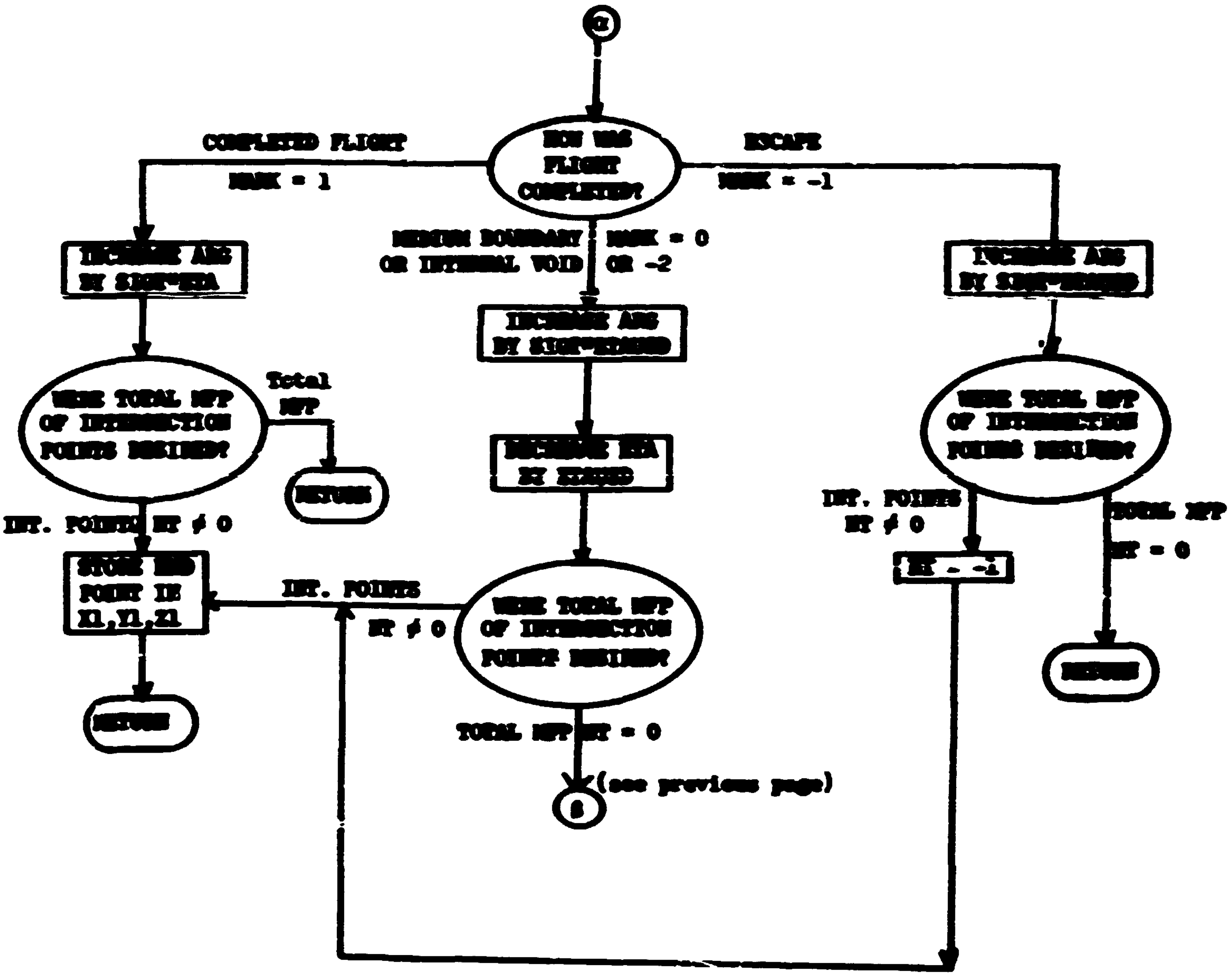
Significant internal variables:

MARK - flag set by GEØM (returned as MRK - defined above)

NINTV - internal flag set to 1 when traversing an internal void (medium 1000).

Limitations: No provision is made in this version for albedo boundaries.


START
WAS CALL FROM INTERMEDIATE BOUNDARY CROSSING?
YES
MARK = 0
NO
MARK = 1
ARG = 0
ETA = P1P2
PUT STARTING POINT IN /GEØMC/
CALL LØØKZ (DETERMINE MEDIUM AND REGION OF STARTING POINT)
β
(from next page)
IS ETA < 0?
YES
RETURN
IS STARTING POINT IN INTERNAL VOID?
NO
YES
NMED = 1000
CALL MSIGTA (DETERMINE TOTAL CROSS SECTION FOR ENERGY IG AND MEDIUM NMED)
SIGT = 0
NINTV = 1
PUT DIRECTION COSINES OF FLIGHT IN /GEØMC/
PUT END POINT IN /GEØMC/
CALL GEØM
WAS FLIGHT JUST COMPLETED IN INTERNAL VOID?
YES
NINTV = 1
CALCULATE DISTANCE TRAVELED IN INTERNAL VOID
NINTV = 0
NO
TRANSFER END POINT TO START POINT IN /GEØMC/
α


HOW WAS FLIGHT COMPLETED?
COMPLETED FLIGHT
MARK = 1
ESCAPE
MARK = -1
MEDIUM BOUNDARY OR INTERNAL VOID
MARK = 0 OR -2
INCREASE ABS BY SIGT*ETA
INCREASE ABS BY SIGT*ETABND
INCREASE ABS BY SIGT*ETABND
DECREASE ETA BY ETABND
WERE TOTAL MFP OF INTERSECTION POINTS DESIRED?
Total MFP
RETURN
INT. POINTS NT ≠ 0
STORE END POINT IN X1,Y1,Z1
RETURN
INT. POINTS
NT ≠ 0
WERE TOTAL MFP OF INTERSECTION POINTS DESIRED?
TOTAL MFP NT = 0
(see previous page)
WERE TOTAL MFP OF INTERSECTION POINTS DESIRED?
INT. POINTS NT ≠ 0
NT = -1
TOTAL MFP NT = 0
RETURN

Subroutine FBANK

Fission neutrons are stored by FBANK in the area in blank common set aside for this purpose. Seven parameters can be stored for NMØST neutrons in this fission bank. If it is called as many as 50 times when the bank is full, HELP and EXIT are called.

Called from: FPRØB.

Subroutines called: HELP, EXIT (library).

Commons required: Blank, NUTRØN, APØLLØ, FISBNK.

Variables required:

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 12)

Variables changed:

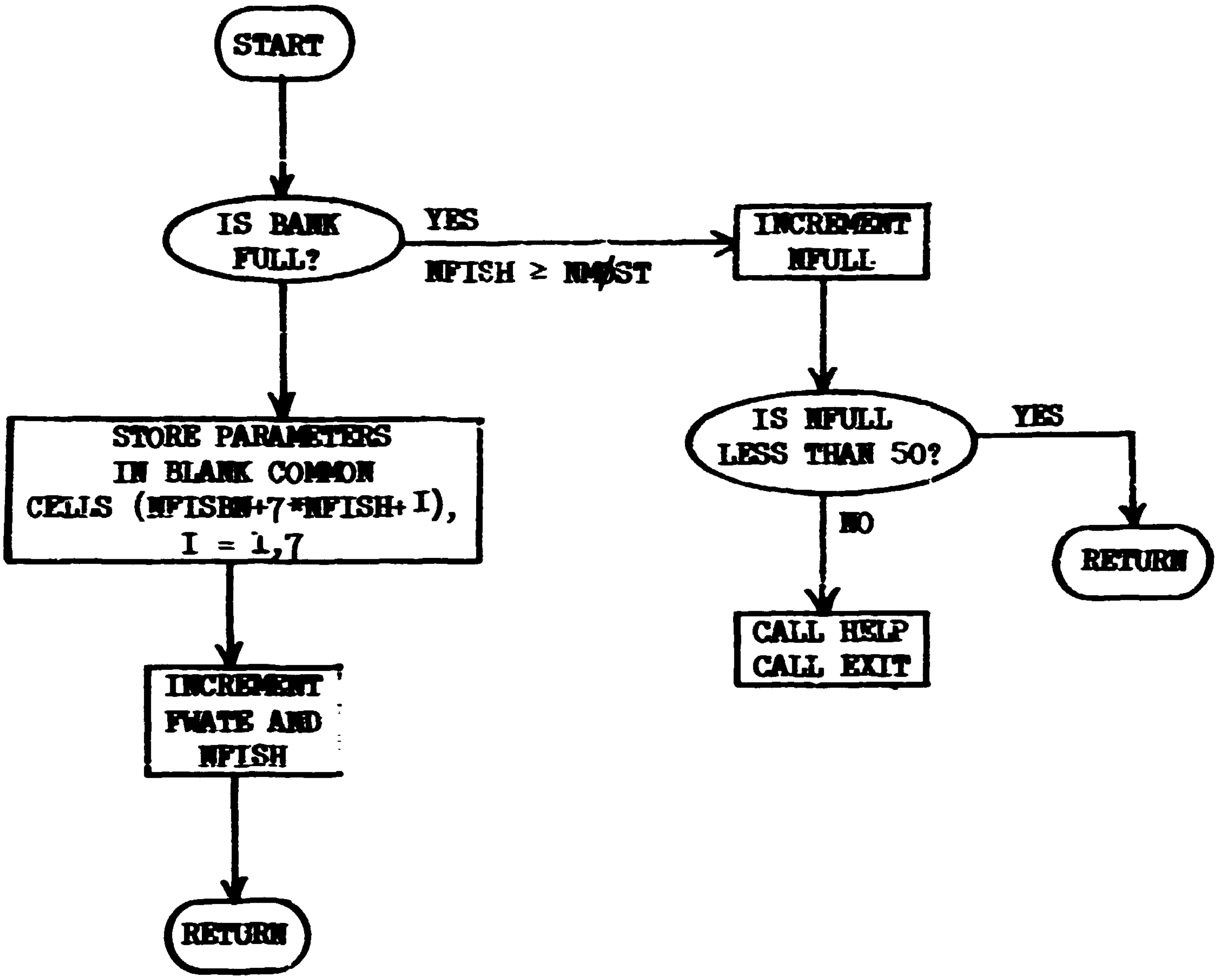
NFISH - incremented after banking,

FWATE - incremented by WATEF after banking.

Significant internal variables:

NFULL - incremented upon each call when bank is full.

Subroutine FBANK



Random Number Package

The random number package is essentially the Ø5R 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 following subprograms are available in the package:

FØRTRAN 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 ØRTSØ (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: $P(R)dR = \left(\frac{4}{T^3\pi}\right)^{1/2} R^{1/2} e^{-R/T} dR$.
R = FISRNF (0)	A neutron speed squared from the Watt fission spectrum: $P(R)\ dR = e^{-R/T} \sinh (2/E^*E_f/T)$, where $T = 0.965 \times 1.913220092 \times 10^{18}$ and $E = 0.533 \times 1.913220092 \times 10^{18}$ (ref. 8).

(cont.)

FØRTRAN Calling Statement	Random Number Generated
CALL RNDIN(R)	Loads R into RANDØM(I), I = 2, 4 if $R \neq 0$. R is read with a Z12 format and must be double precision (8 bytes).
CALL RNDØUT(R)	Loads RANDØM(I), I = 2, 4 into R.

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

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.

Called from: MØRSE.

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

Commons required: Blank, NUTRØN, APØLLØ, FISBNK.

Variables required:

NMED, WATE, NREG, IG (from NUTRØN common, see page 12)

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

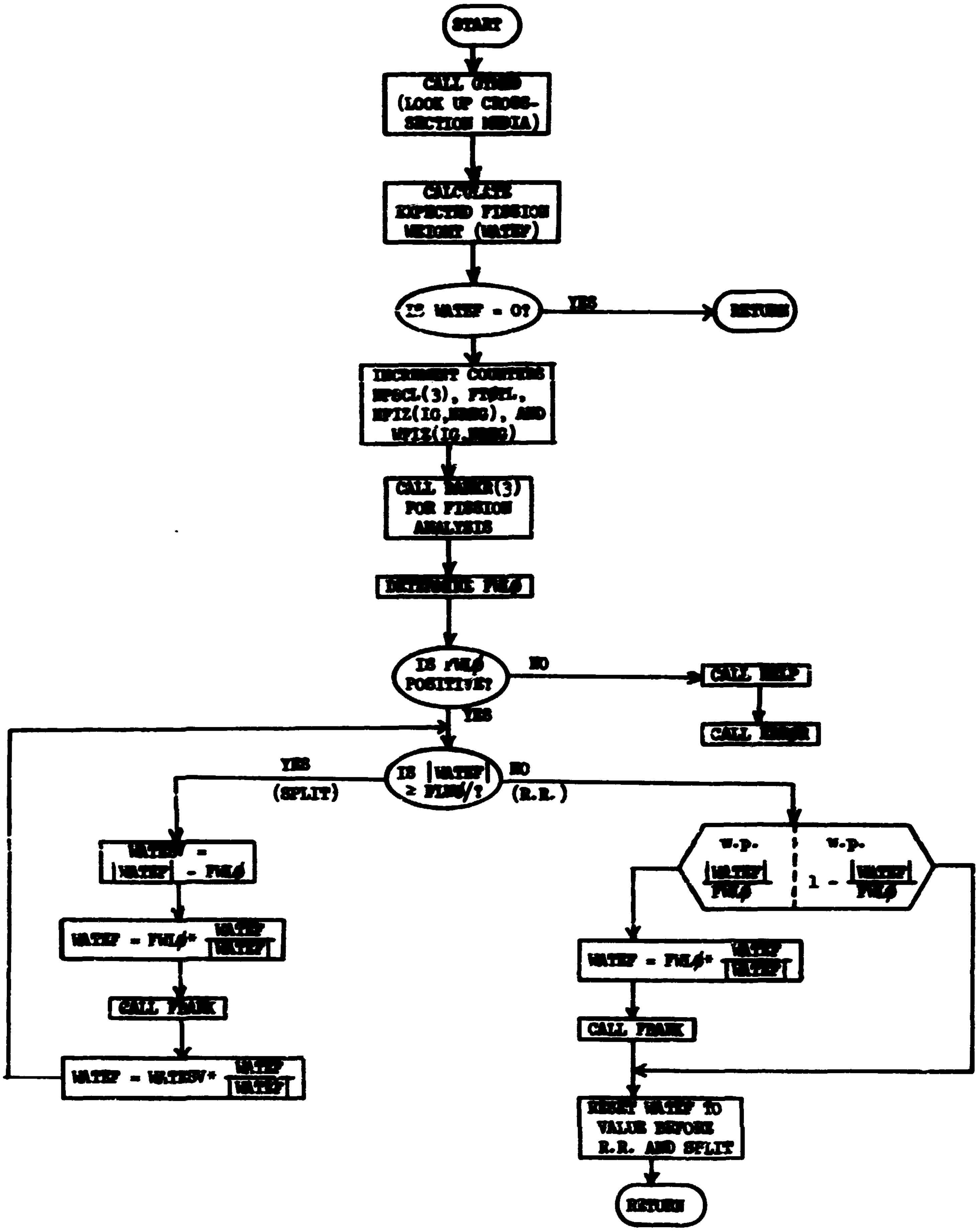
FTØTL

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).


START
CALL
(LOOK UP CROSS-SECTION MEDIA)
CALCULATE EXPECTED FISSION WEIGHT (WATEF)
IS WATEF = 0?
YES
RETURN
INCREMENT COUNTERS NFSCL(3), FISCL, NFIZ(IG,NREG), AND WFIZ(IG,NREG)
CALL BANKR(3) FOR FISSION ANALYSIS
DETERMINE FWLØ
IS FWLØ POSITIVE?
NO
CALL HELP
CALL ERROR
YES
IS |WATEF| ≥ FWLØ/?
YES
(SPLIT)
NO
(R.R.)
WATESV = |WATEF| - FWLØ
WATEF = FWLØ* WATEF/|WATEF|
CALL FBANK
WATEF = WATESV* WATEF/|WATEF|
w.p. |WATEF|/FWLØ
w.p. 1 - |WATEF|/FWLØ
WATEF = FWLØ* WATEF/|WATEF|
CALL FBANK
RESET WATEF TO VALUE BEFORE R.R. AND SPLIT
RETURN

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 Nth 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.

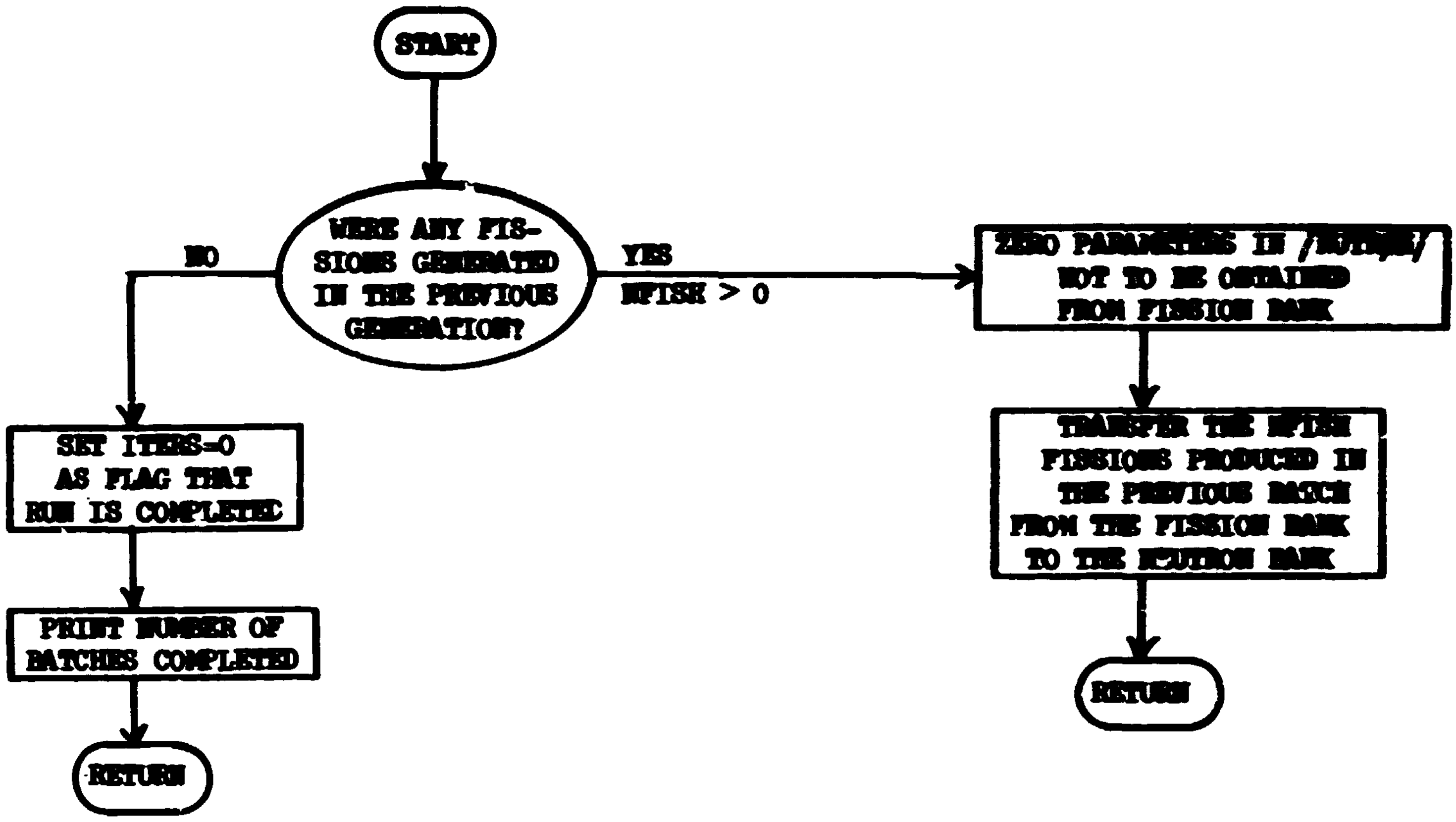
Variables changed:

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 12)
X, Y, Z WATE AGE IG NAMEX	set to values found in fission bank (in NUTRØN common, see page 12) Note: IG is group index of neutron causing fission.

Subroutine FSØUR


START
WERE ANY FISSIONS GENERATED IN THE PREVIOUS GENERATION?
NO
YES
NFISH > 0
SET ITERS=0 AS FLAG THAT RUN IS COMPLETED
PRINT NUMBER OF BATCHES COMPLETED
RETURN
ZERO PARAMETERS IN /NUTRØN/ NOT TO BE OBTAINED FROM FISSION BANK
TRANSFER THE NFISH FISSIONS PRODUCED IN THE PREVIOUS BATCH FROM THE FISSION BANK TO THE NEUTRON BANK
RETURN

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[9-11] 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

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

where η is the distance traveled in mean-free paths. Selection of a particular flight path ETA from $P_o(\eta)$ is done by the function EXPRNF (in random number package, see page 34).

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

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

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 $P_o(\eta)$. 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

$$\mathrm{ETA} = \mathrm{BIAS}{*}\mathrm{ETA}' .$$

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

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

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

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

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

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

where $\eta = \mathrm{BIAS}{*}\eta'$. The path-length ETA is then given by

$$\mathrm{ETA} = \mathrm{BIAS}{*}\mathrm{ETA}',$$

and the particle's weight multiplied by the factor

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

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

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

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.

Commons required: Blank, NUTRØN, APØLLØ.

Variables required:

IG, X, Y, Z, U, V, W, WATE, NREG (from NUTRØN common, see page 12)

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).

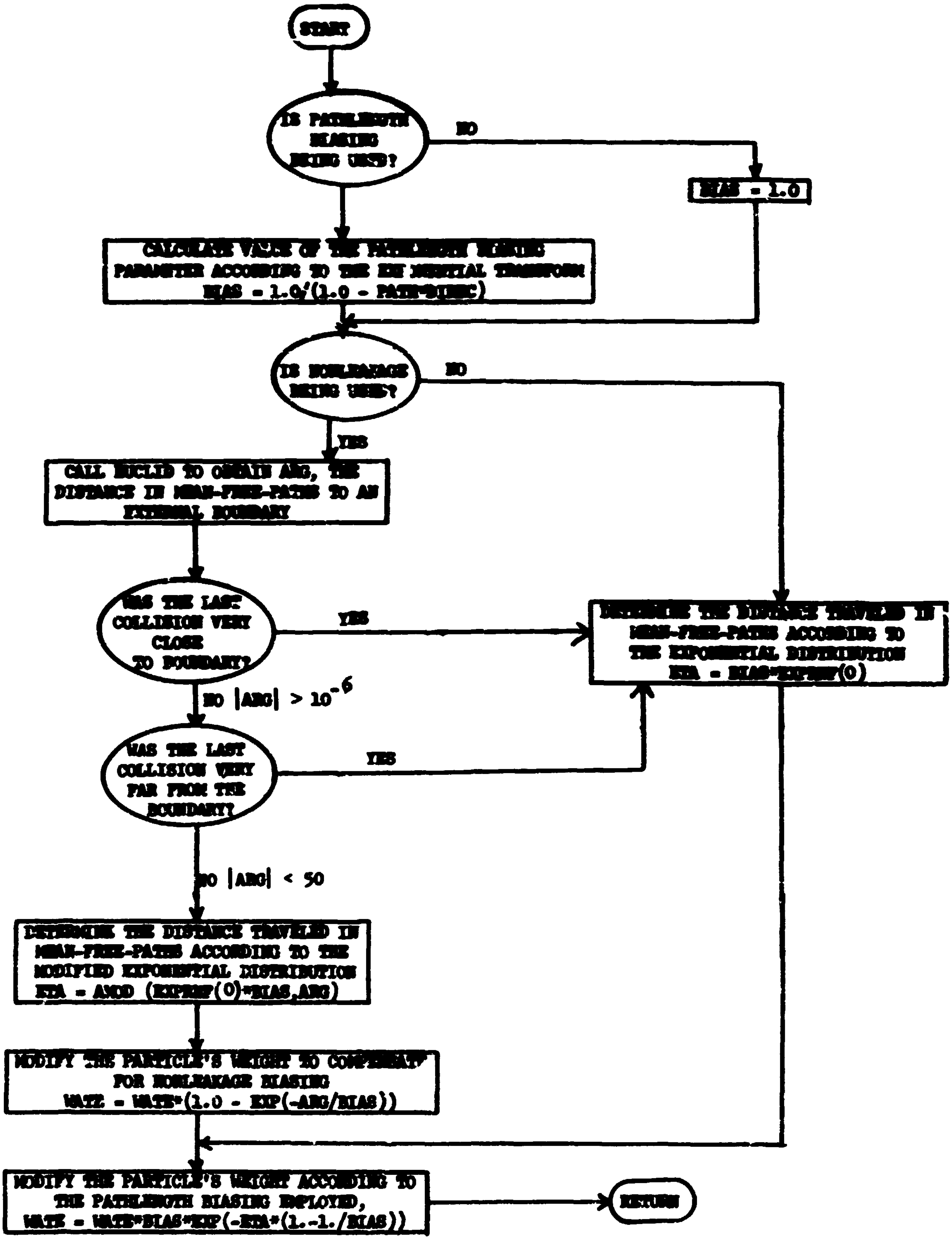
Variables changed:

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.


START
IS PATHLENGTH BIASING BEING USED?
NO
BIAS = 1.0
CALCULATE VALUE OF THE PATHLENGTH BIASING PARAMETER ACCORDING TO THE EXPONENTIAL TRANSFORM
BIAS = 1.0/(1.0 - PATH*DIREC)
IS NONLEAKAGE BEING USED?
NO
YES
CALL EUCLID TO OBTAIN ARG, THE DISTANCE IN MEAN-FREE-PATHS TO AN EXTERNAL BOUNDARY
WAS THE LAST COLLISION VERY CLOSE TO BOUNDARY?
YES
DETERMINE THE DISTANCE TRAVELED IN MEAN-FREE-PATHS ACCORDING TO THE EXPONENTIAL DISTRIBUTION
ETA = BIAS*EXPRNF(0)
NO |ARG| > 10^-6
WAS THE LAST COLLISION VERY FAR FROM THE BOUNDARY?
YES
NO |ARG| < 50
DETERMINE THE DISTANCE TRAVELED IN MEAN-FREE-PATHS ACCORDING TO THE MODIFIED EXPONENTIAL DISTRIBUTION
ETA = AMOD (EXPRNF(0)*BIAS,ARG)
MODIFY THE PARTICLE'S WEIGHT TO COMPENSATE FOR NONLEAKAGE BIASING
WATE = WATE*(1.0 - EXP(-ARG/BIAS))
MODIFY THE PARTICLE'S WEIGHT ACCORDING TO THE PATHLENGTH BIASING EMPLOYED,
WATE = WATE*BIAS*EXP(-ETA*(1.-1./BIAS))
RETURN

Subroutine GETNT(N)

Three entry points are used in this routine. Entry SETNT saves the address (in words) of the first ce' vailable 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 Nth 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: INPUT (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 Fig. 3. Notice that IG, NAME, NAMEX, NMED, and NREG are stored in 2-byte words (and are therefore limited to ≤ 65535), symbolized by a dotted line splitting the normal 4-byte word.

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 12)

Variables changed:

SETNT: NLAST

GETNT: variables in common NUTRØN required by SETNT above,

STØRNT: 12 consecutive locations in blank common.

Significant internal variables:

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

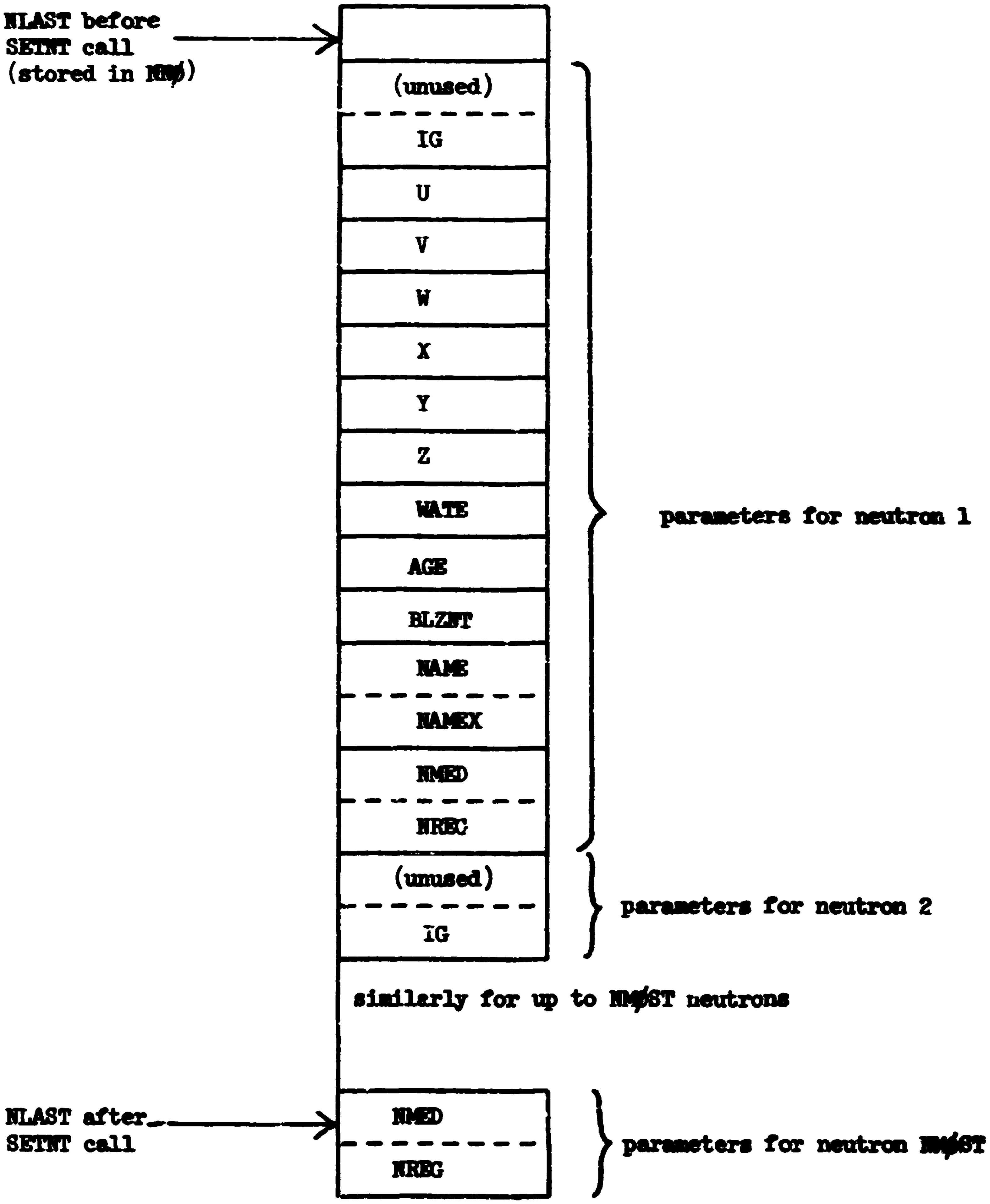
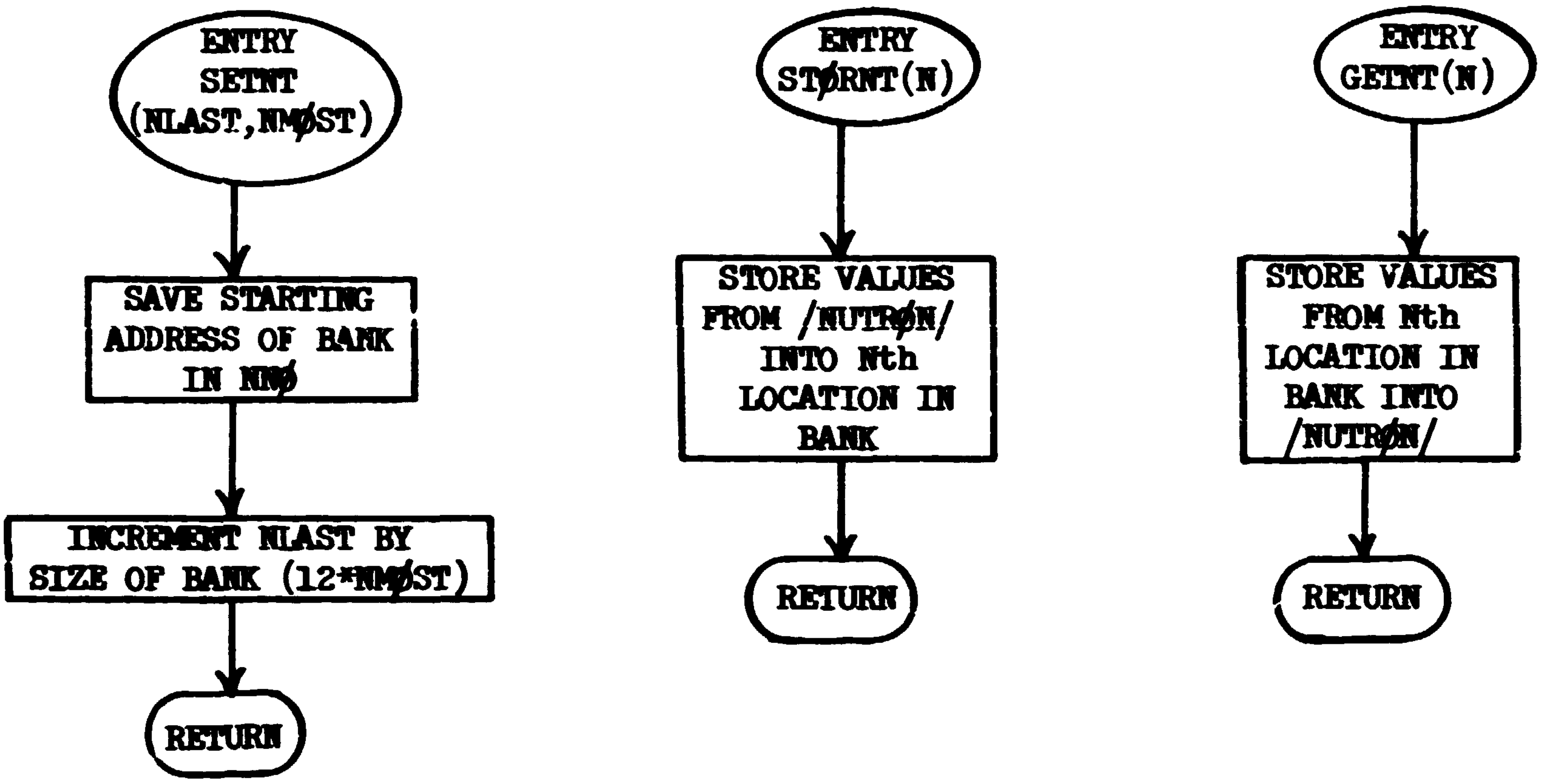


Fig. 3. Layout of the Neutron Bank in Blank Common

Subroutine GETNT(N)


ENTRY
SETNT
(NLAST,NMØST)
SAVE STARTING ADDRESS OF BANK IN NNØ
INCREMENT NLAST BY SIZE OF BANK (12*NMØST)
RETURN
ENTRY
STØRNT(N)
STORE VALUES FROM /NUTRØN/ INTO Nth LOCATION IN BANK
RETURN
ENTRY
GETNT(N)
STORE VALUES FROM Nth LOCATION IN BANK INTO /NUTRØN/
RETURN

Subroutine GØMST (TSIG, MARK)

The end-of-flight coordinates are computed assuming the starting medium extends infinitely. The proper data are stored in GEØMC before calling GEØM and is restored after the GEØM call. If the flight is starting in interior void (NMED = 1000), velocity components (or direction cosines) rather than an end point are given to GEØM. If an albedo medium is encountered the flag NALB is set to the albedo medium number, and then NØRML and GØMFLP are called, respectively, to determine the normal to the surface encountered, and to reset certain parameters for GEØM to use later in going away from the albedo surface.

Called from: NXTCØL.

Subroutines called: GEØM, NØRML.

Functions used: SQRT (library)

Commons required: APØLLØ, NUTRØN, GEØMC

Variables required:

XØLD, YØLD, ZØLD, NMED,
U, V, W, NREG } from NUTRØN common, see page 12)

IBLZØ - packed word containing block and zone numbers for starting point,

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

MARK - initial value of flag used by GEØM,

TSIG - total cross section of starting point medium,

ETA - flight distance in m.f.p.

Variables changed:

X, Y, Z - end point of flight,

NALB - albedo flag (= MEDALB or 0),

MARK - flag indicating type of termination of flight,

ETA - actual flight distance (in m.f.p.) if albedo collision occurs,

NMED - medium of end point,

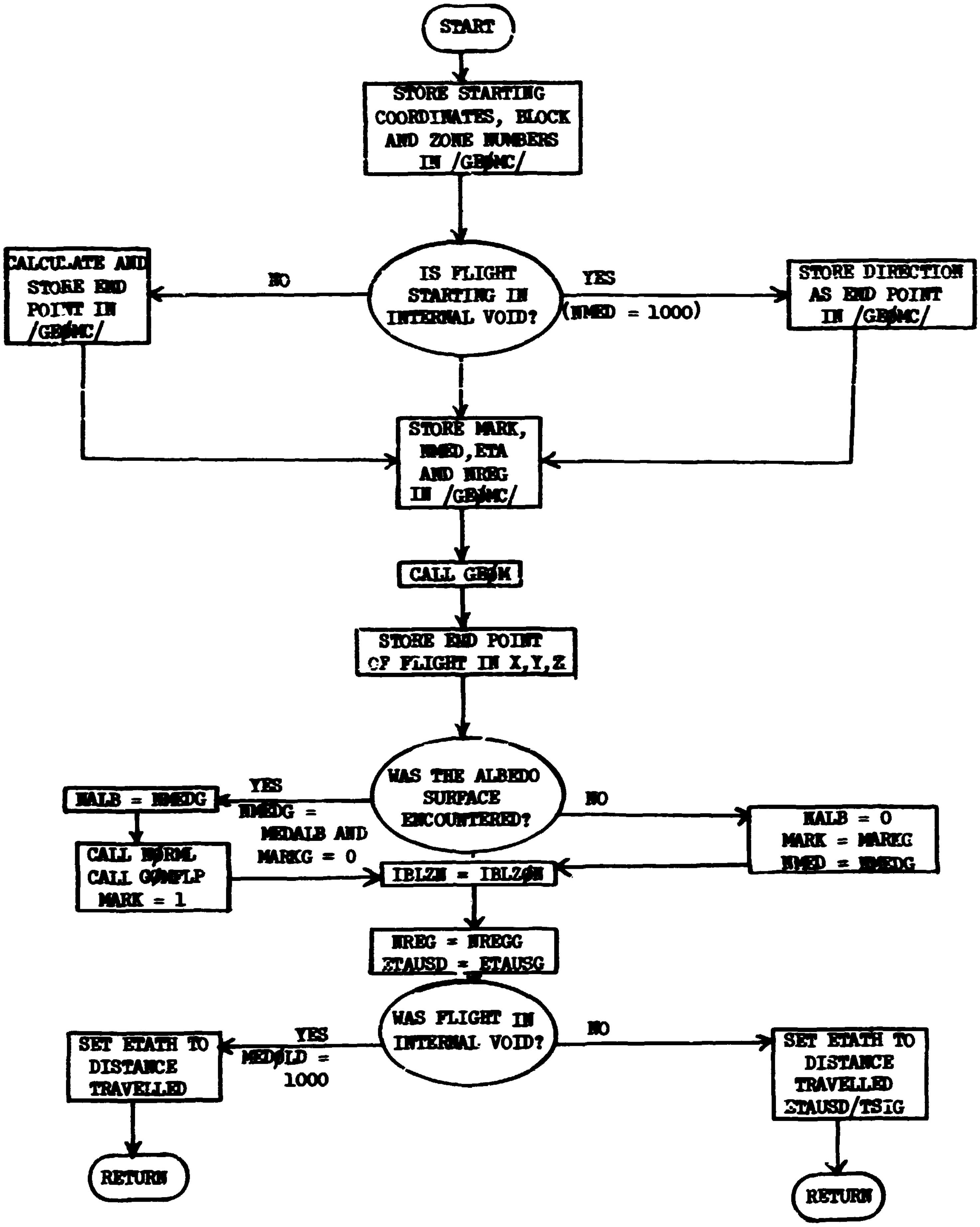
NREG - region of end point,

ETAU3D - actual flight distance (in m.f.p.),

IBLZN - block and zone of end point,

ETATH - actual flight distance (in cm).

Subroutine GØMST (TSIG,MARK)


START
STORE STARTING COORDINATES, BLOCK AND ZONE NUMBERS IN /GEØMC/
IS FLIGHT STARTING IN INTERNAL VOID?
NO
CALCULATE AND STORE END POINT IN /GEØMC/
YES
(NMED = 1000)
STORE DIRECTION AS END POINT IN /GEØMC/
STORE MARK, NMED, ETA AND NREG IN /GEØMC/
CALL GEØM
STORE END POINT OF FLIGHT IN X,Y,Z
WAS THE ALBEDO SURFACE ENCOUNTERED?
YES
NMEDG = MEDALB AND MARKG = 0
NALB = NMEDG
CALL NØRML
CALL GØMFLP
MARK = 1
NO
NALB = 0
MARK = MARKG
NMED = NMEDG
IBLZN = IBLZØN
NREG = NREGG
ETAUSD = ETAUSG
WAS FLIGHT IN INTERNAL VOID?
YES
MEDØLD = 1000
SET ETATH TO DISTANCE TRAVELLED
RETURN
NO
SET ETATH TO DISTANCE TRAVELLED ETAUSD/TSIG
RETURN

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 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.

Another version of GPRØb which has been found to be more useful in some cases does not use GWL as a desired gamma weight but instead uses it as the probability of generating a gamma ray. Thus, a random number, if 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.

Called from: MØRSE

Subroutines called: GAMGEN, GSTØRE, HELP, ERRØR.

Functions used: SIGN, ABS (library).

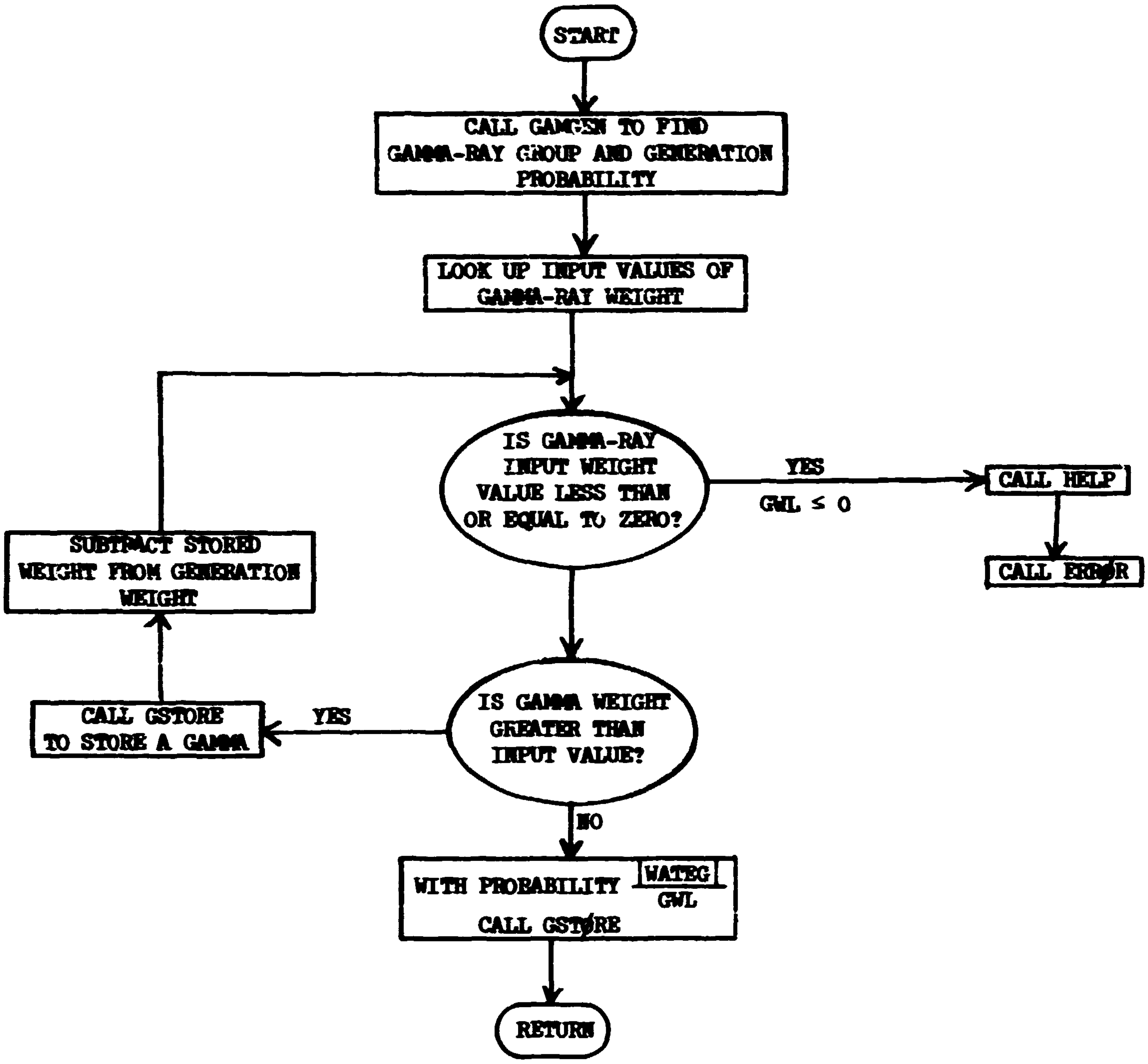
Commons required: Blank, NUTRØN, APØLLØ.

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 GPRØB


START
CALL GAMGEN TO FIND GAMMA-RAY GROUP AND GENERATION PROBABILITY
LOOK UP INPUT VALUES OF GAMMA-RAY WEIGHT
IS GAMMA-RAY INPUT WEIGHT VALUE LESS THAN OR EQUAL TO ZERO?
YES
GWL ≤ 0
CALL HELP
CALL ERRØR
SUBTRACT STORED WEIGHT FROM GENERATION WEIGHT
CALL GSTORE TO STORE A GAMMA
YES
IS GAMMA WEIGHT GREATER THAN INPUT VALUE?
NO
WITH PROBABILITY WATEG / GWL
CALL GSTØRE
RETURN

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 } (from NUTRØN common, see page 12)
NAME
U,V,W

Variables changed:

NMEM - last location in bank that has been used,

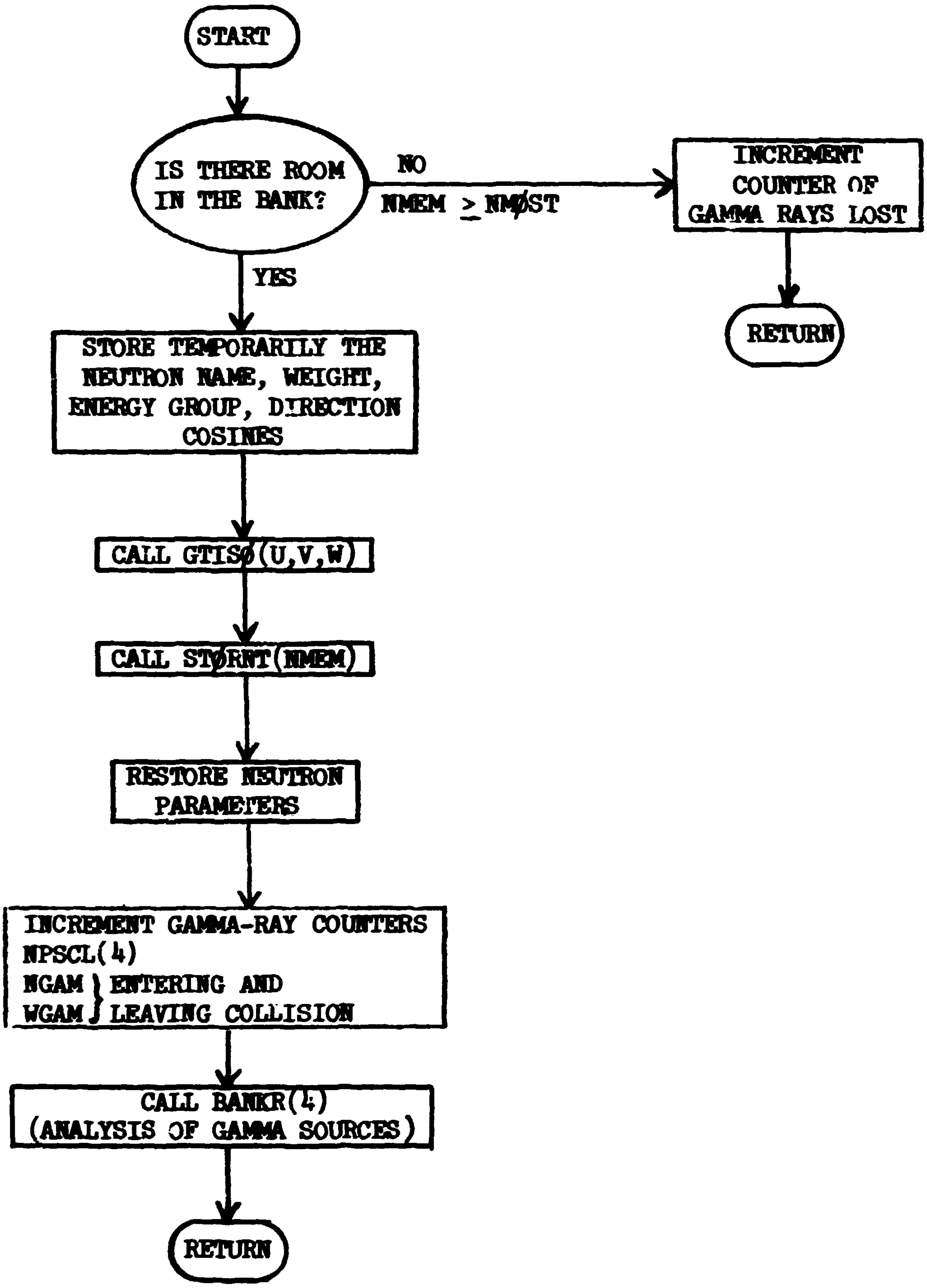
NEWNM - the gamma-ray name.

Significant internal variables:

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

Limitations: Isotropic gamma-ray emission.

Subroutine GSTØRE



START
IS THERE ROOM IN THE BANK?
NO
NMEM ≥ NMØST
INCREMENT COUNTER OF GAMMA RAYS LOST
RETURN
YES
STORE TEMPORARILY THE NEUTRON NAME, WEIGHT, ENERGY GROUP, DIRECTION COSINES
CALL GTISØ(U,V,W)
CALL STØRNT(NMEM)
RESTORE NEUTRON PARAMETERS
INCREMENT GAMMA-RAY COUNTERS
NPSCL(4)
NGAM } ENTERING AND
WGAM } LEAVING COLLISION
CALL BANKR(4)
(ANALYSIS OF GAMMA SOURCES)
RETURN

Subroutine INPJT

The basic functions of subroutine INPUT are to 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. For complete details, refer to the flow chart.

Called from: MØRSE.

Subroutines called:

DATE - provides EBCDIC string containing day of week and date,

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,

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.

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)

MODEL - (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 8, 12, 167. A more detailed listing of input is given in Appendix C.)

CARD A (20A4)

Title

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

CARD B (14I5)

NSTRT, NMØST, NITS, NQUIT, NGPQTN, NGPQTG, NMGP, NMTG, NCØLTP, IADJM, MAXTIM, 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

CARDS E1 (7E10.4) (skipped if ISØUR > 0) (read by SØRIN)

FS(I), I = 1, NGPFS

CARDS E2 (7E10.4) (skipped if ISØUR > 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, NDRG, NRG2, WTHIH1, WTLØW1, WTAVE1, PATH

(read until NGP1 < 0)

CARDS K (7E10.4) (skipped if IEBIAS ≤ 0)

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

CARD L (14I5)

NSØUR, MFISTP, NKCALC, NØRMF

CARDS M (7E10.4) (skipped if MFISTP ≤ 0)

(FWLØ(I),I=1,MXREG)

CARDS N (7E10.4) (skipped if MFISTP ≤ 0)

(FSE(IG,IMED),IG=1,NMGP),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

XSEC called for cross-section data

SCØRIN called for analysis 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 (48484848_{16}),

All in common APØLLØ - except I1, I0, ITIME, NLAST are filled with junk word,

MAXTIM

DFF
NGPQT0
NGPQT1
NGPQT2
NGPQT3
NWPCØL

} for definitions, see common APØLLØ, page 8, and diagram of energy group structure, page 11

NCØLPR

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

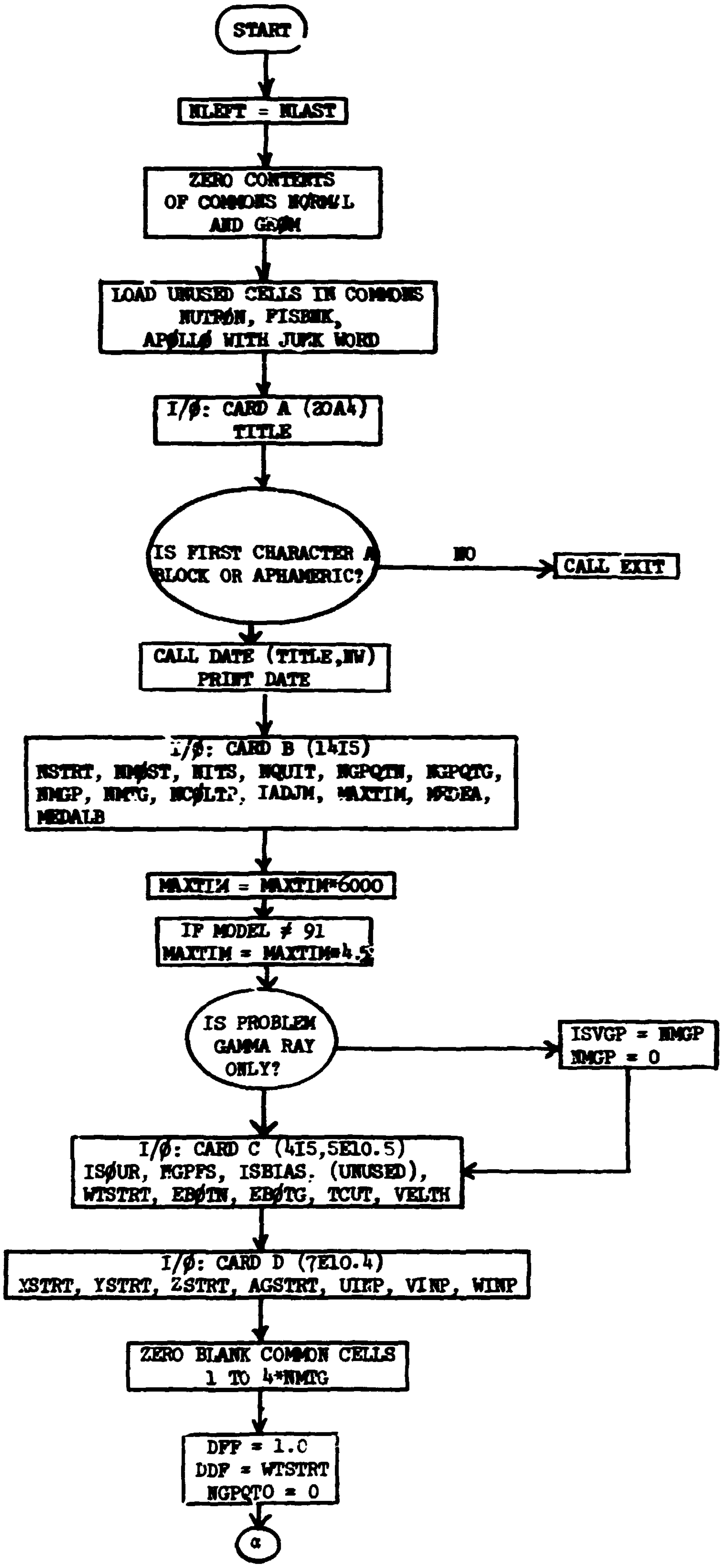
LØCFWL
LØCEPR
LØCNSC
LØCFSN
NGEØM
NLAST

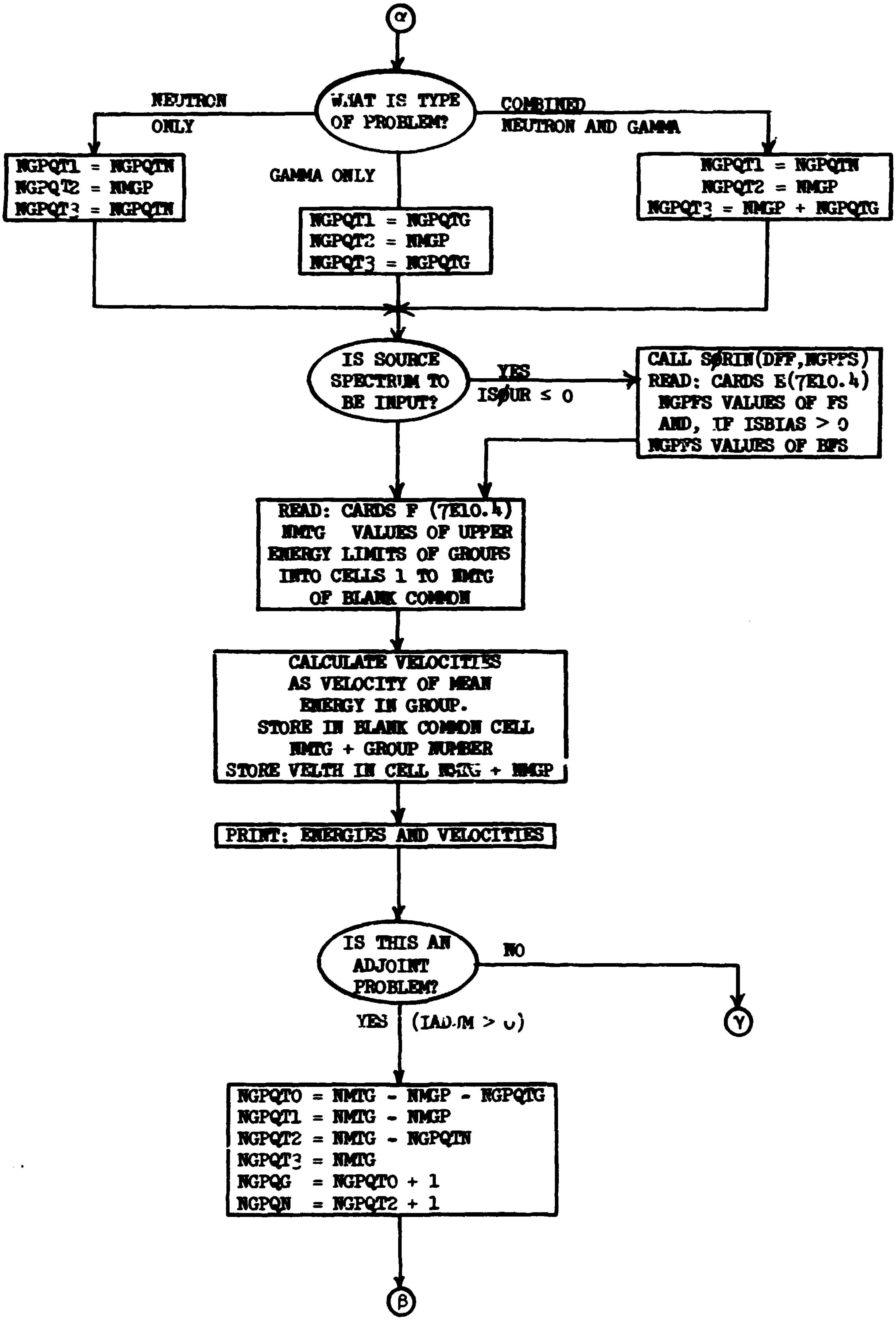
} for definitions, see common APØLLØ, page 8

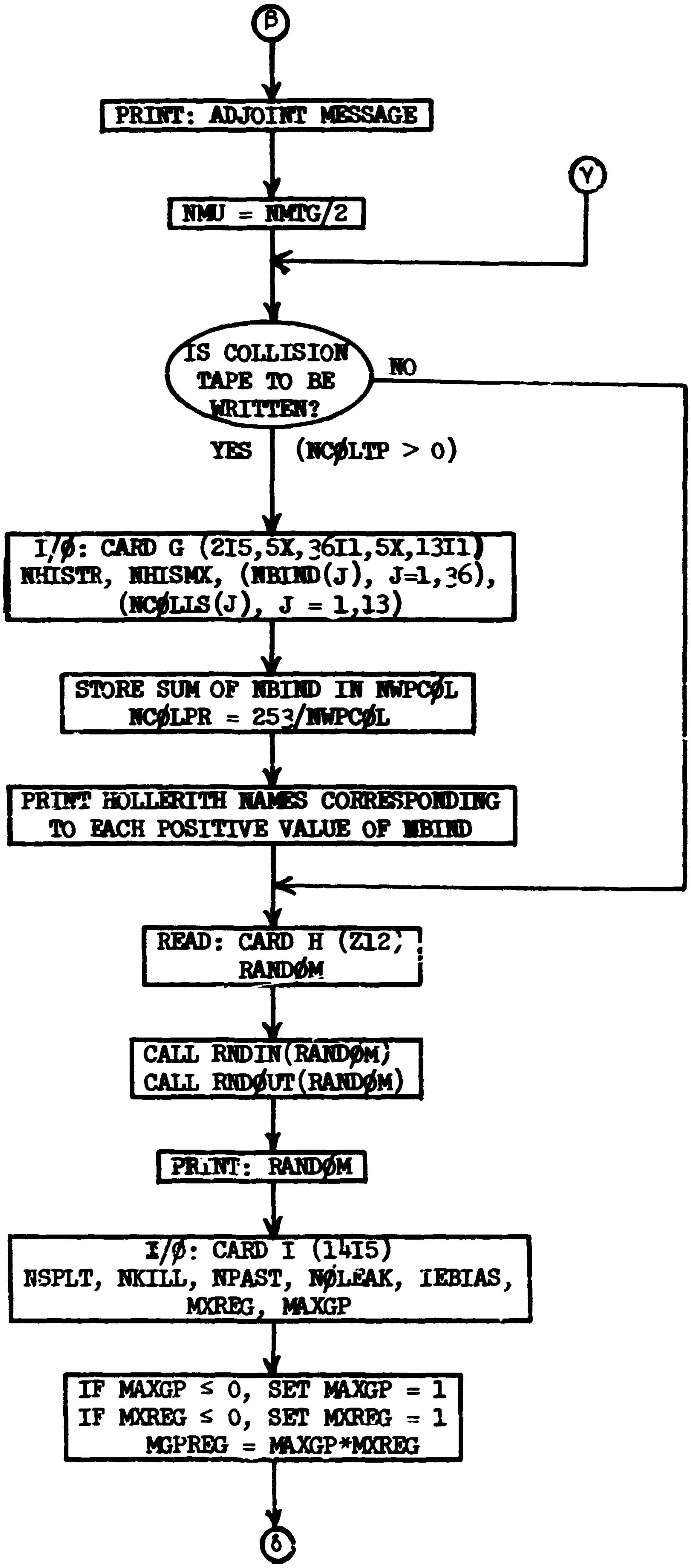
NSIGL

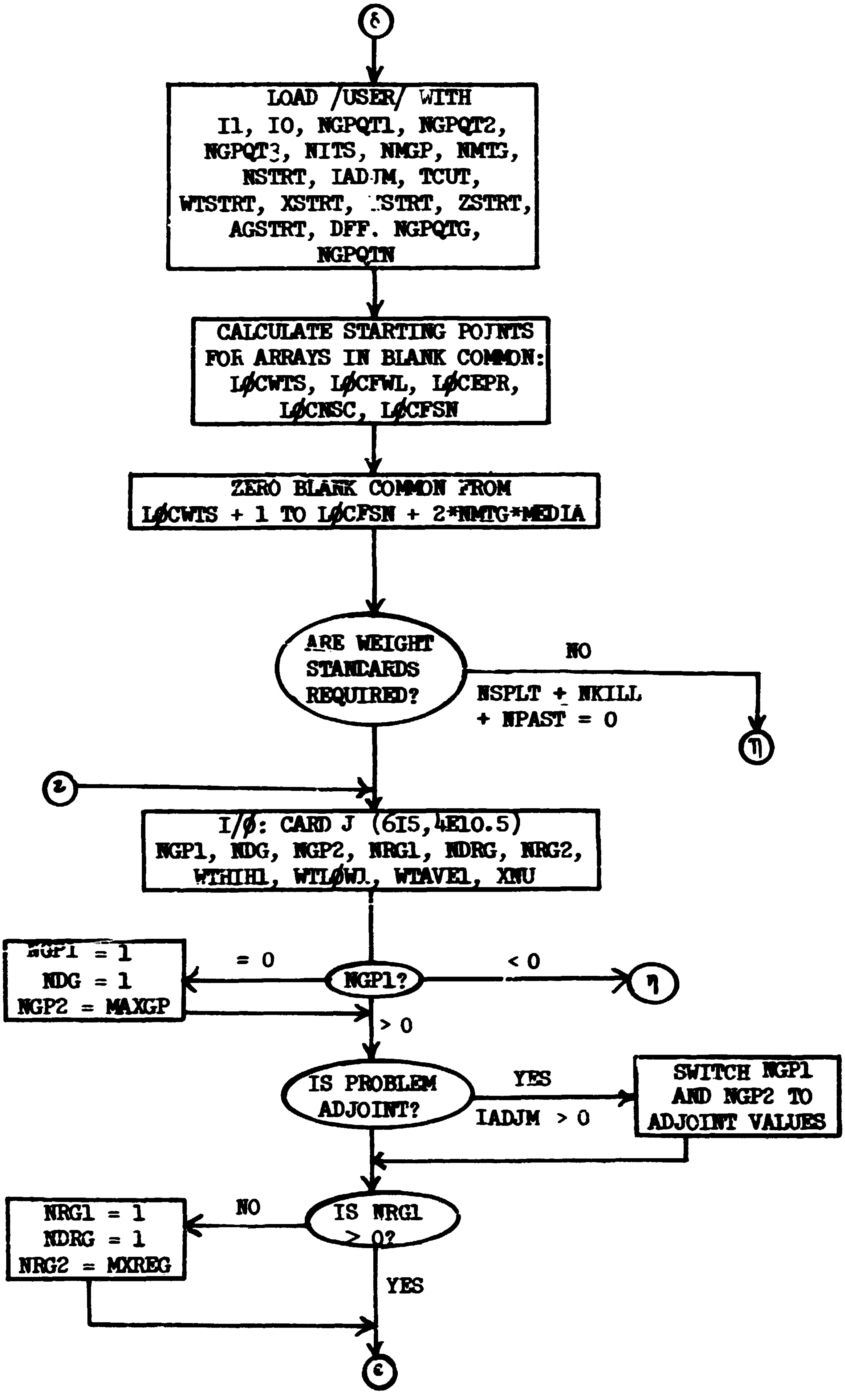
NFISBN

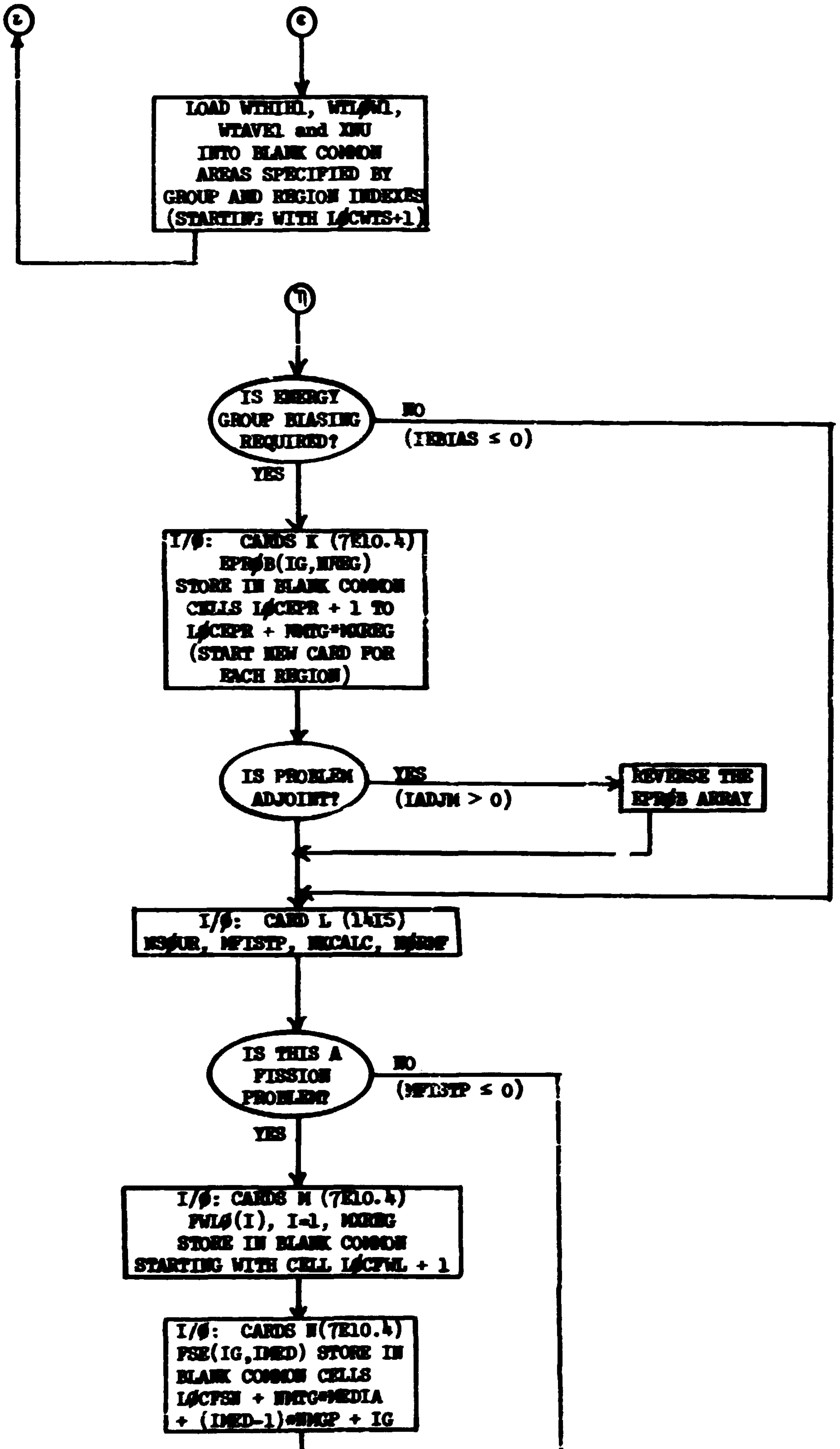
Subroutine INPUT

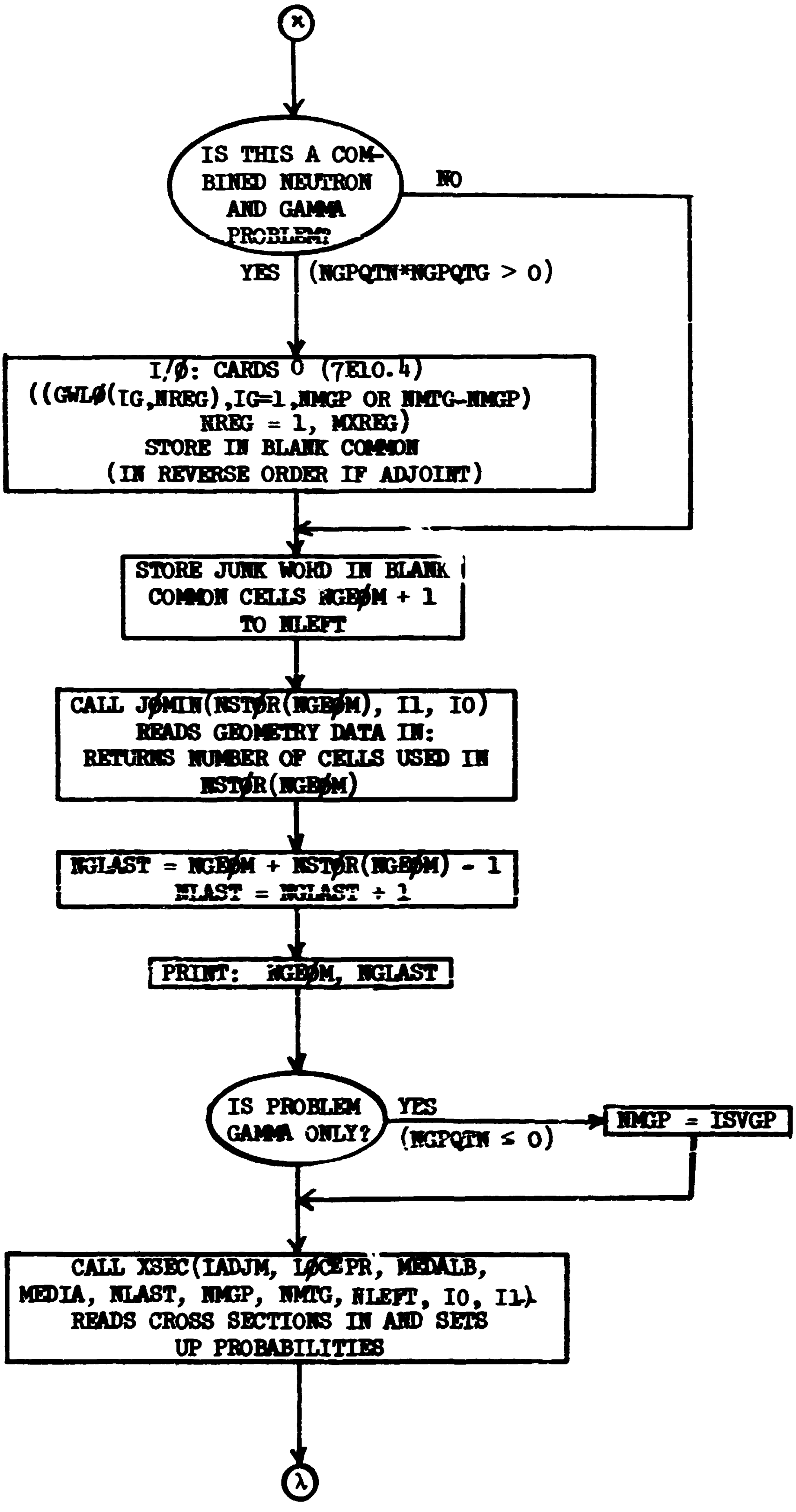


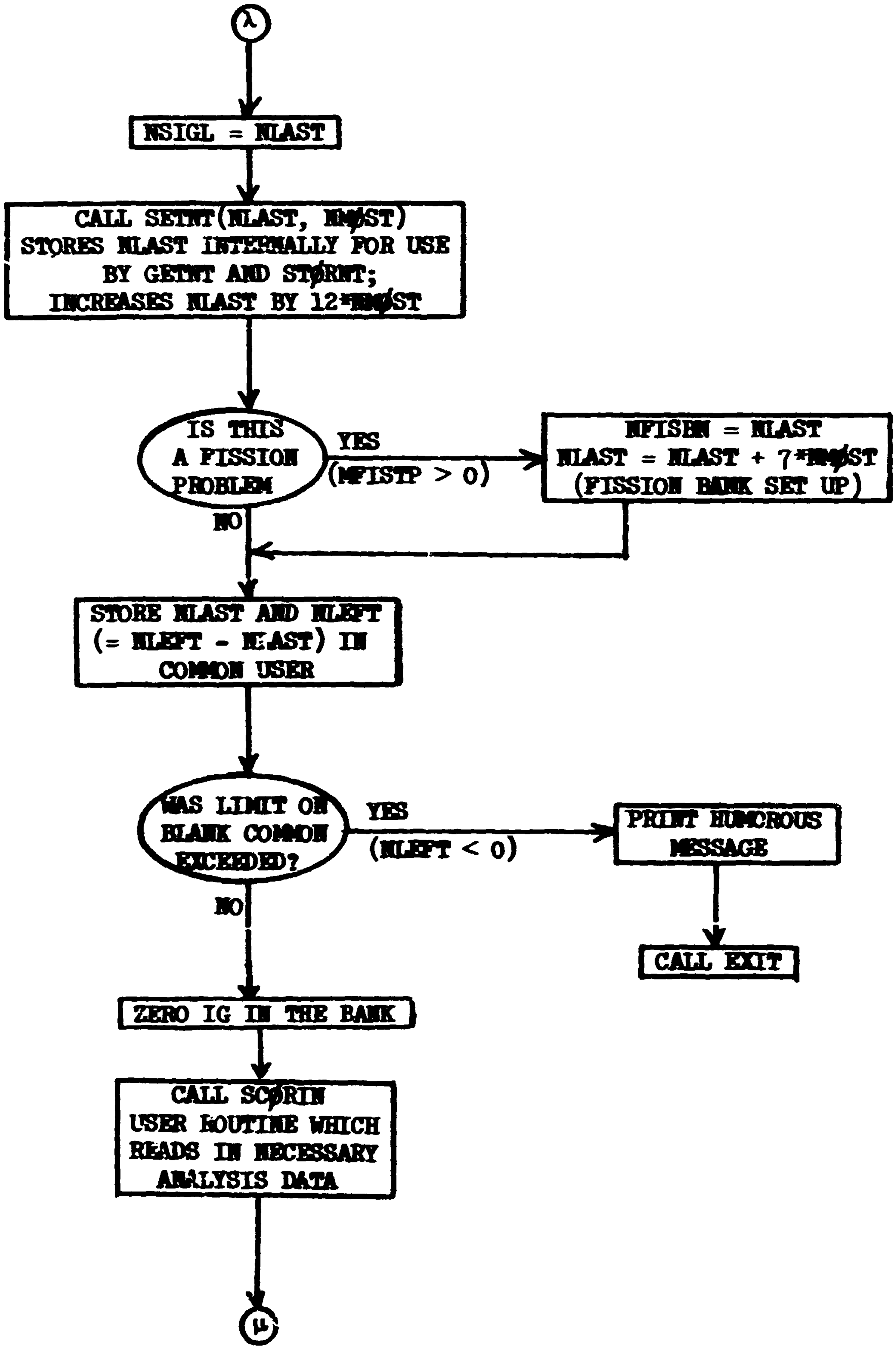

α
WHAT IS TYPE OF PROBLEM?
NEUTRON ONLY
COMBINED NEUTRON AND GAMMA
GAMMA ONLY
NGPQT1 = NGPQTN
NGPQT2 = NMGP
NGPQT3 = NGPQTN
NGPQT1 = NGPQTG
NGPQT2 = NMGP
NGPQT3 = NGPQTG
NGPQT1 = NGPQTN
NGPQT2 = NMGP
NGPQT3 = NMGP + NGPQTG
IS SOURCE SPECTRUM TO BE INPUT?
YES
ISØUR ≤ 0
CALL SØRIN(DFF,NGPFS)
READ: CARDS E(7E10.4)
NGPFS VALUES OF FS
AND, IF ISBIAS > 0
NGPFS VALUES OF BFS
READ: CARDS F (7E10.4)
NMTG VALUES OF UPPER
ENERGY LIMITS OF GROUPS
INTO CELLS 1 TO NMTG
OF BLANK COMMON
CALCULATE VELOCITIES
AS VELOCITY OF MEAN
ENERGY IN GROUP.
STORE IN BLANK COMMON CELL
NMTG + GROUP NUMBER
STORE VELTH IN CELL NMTG + NMGP
PRINT: ENERGIES AND VELOCITIES
IS THIS AN ADJOINT PROBLEM?
NO
γ
YES
(IADJM > 0)
NGPQT0 = NMTG - NMGP - NGPQTG
NGPQT1 = NMTG - NMGP
NGPQT2 = NMTG - NGPQTN
NGPQT3 = NMTG
NGPQG = NGPQT0 + 1
NGPQN = NGPQT2 + 1
β

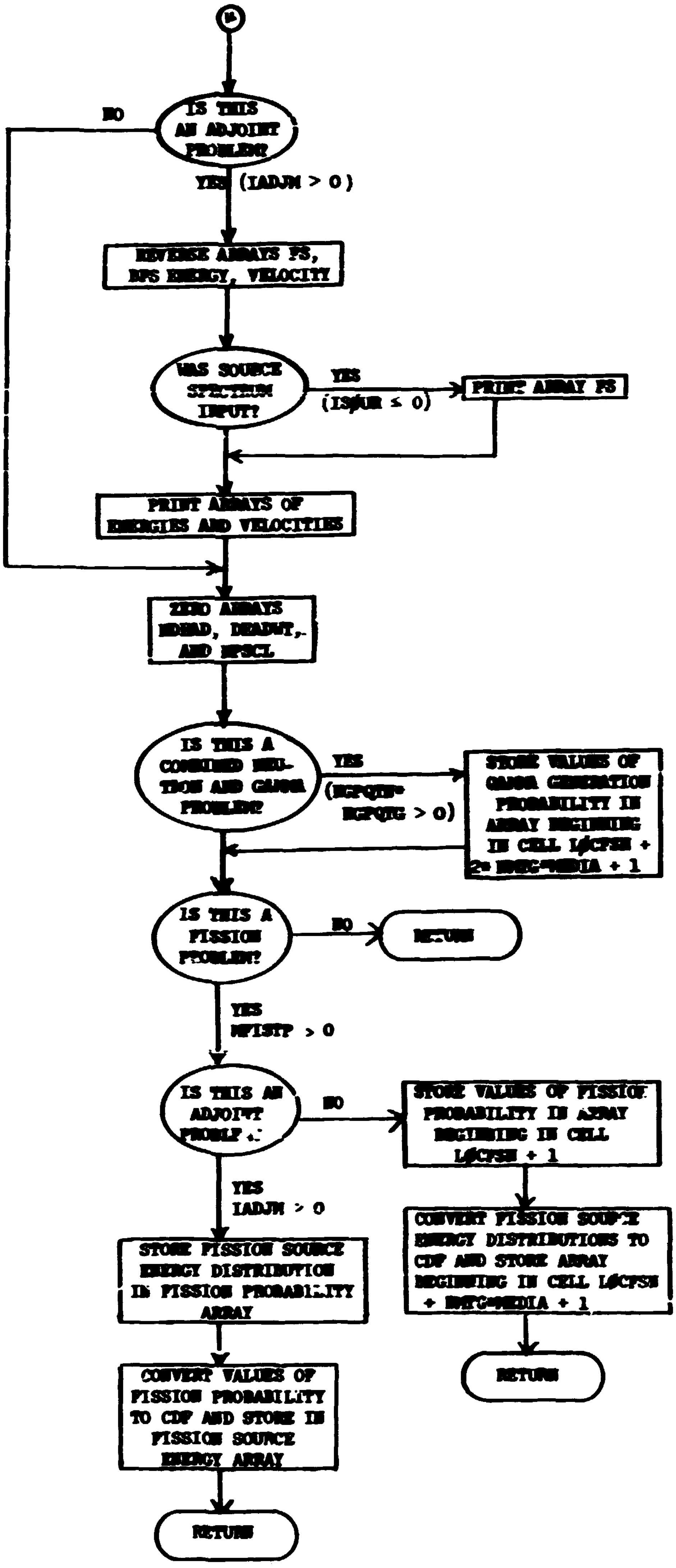

β
PRINT: ADJOINT MESSAGE
γ
NMU = NMTG/2
IS COLLISION TAPE TO BE WRITTEN?
NO
YES
(NCØLTP > 0)
I/Ø: CARD G (2I5,5X,36I1,5X,13I1)
NHISTR, NHISMX, (NBIND(J), J=1,36), (NCØLLS(J), J = 1,13)
STORE SUM OF NBIND IN NWPCØL
NCØLPR = 253/NWPCØL
PRINT HOLLERITH NAMES CORRESPONDING TO EACH POSITIVE VALUE OF NBIND
READ: CARD H (Z12)
RANDØM
CALL RNDIN(RANDØM)
CALL RNDØUT(RANDØM)
PRINT: RANDØM
I/Ø: CARD I (14I5)
NSPLT, NKILL, NPAST, NØLEAK, IEBIAS, MXREG, MAXGP
IF MAXGP ≤ 0, SET MAXGP = 1
IF MXREG ≤ 0, SET MXREG = 1
MGPREG = MAXGP*MXREG
δ


8
LOAD /USER/ WITH
I1, IO, NGPQT1, NGPQT2,
NGPQT3, NITS, NMGP, NMTG,
NSTRT, IADJM, TCUT,
WTSTRT, XSTRT, YSTRT, ZSTRT,
AGSTRT, DFF. NGPQTG,
NGPQTN
CALCULATE STARTING POINTS
FOR ARRAYS IN BLANK COMMON:
LØCWTS, LØCFWL, LØCEPR,
LØCNSC, LØCFSN
ZERO BLANK COMMON FROM
LØCWTS + 1 TO LØCFSN + 2*NMTG*MEDIA
ARE WEIGHT
STANDARDS
REQUIRED?
NO
NSPLT + NKILL
+ NPAST = 0
7
2
I/Ø: CARD J (6I5,4E10.5)
NGP1, NDG, NGP2, NRG1, NDRG, NRG2,
WTHIH1, WTLØW1, WTAVE1, XNU
NGP1 = 1
NDG = 1
NGP2 = MAXGP
= 0
NGP1?
< 0
7
> 0
IS PROBLEM
ADJOINT?
YES
IADJM > 0
SWITCH NGP1
AND NGP2 TO
ADJOINT VALUES
NRG1 = 1
NDRG = 1
NRG2 = MXREG
NO
IS NRG1
> 0?
YES
c


6
LOAD WTHIE1, WTLØW1, WTAVE1 and XNU INTO BLANK COMMON AREAS SPECIFIED BY GROUP AND REGION INDEXES (STARTING WITH LØCWTS+1)
7
IS ENERGY GROUP BIASING REQUIRED?
NO (IEBIAS ≤ 0)
YES
I/Ø: CARDS K (7E10.4) EPRØB(IG,NREG) STORE IN BLANK COMMON CELLS LØCEPR + 1 TO LØCEPR + NMTG*MXREG (START NEW CARD FOR EACH REGION)
IS PROBLEM ADJOINT?
YES (IADJM > 0)
REVERSE THE EPRØB ARRAY
I/Ø: CARD L (14I5) NSØUR, NFISTP, NKCALC, NØRMF
IS THIS A FISSION PROBLEM?
NO (NFISTP ≤ 0)
YES
I/Ø: CARDS M (7E10.4) FWLØ(I), I=1, MXREG STORE IN BLANK COMMON STARTING WITH CELL LØCFWL + 1
I/Ø: CARDS N(7E10.4) FSE(IG,IMED) STORE IN BLANK COMMON CELLS LØCFSN + NMTG*MEDIA + (IMED-1)*NMGP + IG


x
IS THIS A COMBINED NEUTRON AND GAMMA PROBLEM?
NO
YES
(NGPQTN*NGPQTG > 0)
I/Ø: CARDS O (7E10.4)
((GWLØ(IG,NREG),IG=1,NMGP OR NMTG-NMGP)
NREG = 1, MXREG)
STORE IN BLANK COMMON
(IN REVERSE ORDER IF ADJOINT)
STORE JUNK WORD IN BLANK COMMON CELLS NGEØM + 1 TO NLEFT
CALL JØMIN(NSTØR(NGEØM), I1, IO)
READS GEOMETRY DATA IN:
RETURNS NUMBER OF CELLS USED IN NSTØR(NGEØM)
NGLAST = NGEØM + NSTØR(NGEØM) - 1
NLAST = NGLAST + 1
PRINT: NGEØM, NGLAST
IS PROBLEM GAMMA ONLY?
YES
(NGPQTN ≤ 0)
NMGP = ISVGP
CALL XSEC(IADJM, LØCEPR, MEDALB, MEDIA, NLAST, NMGP, NMTG, NLEFT, IO, I1)
READS CROSS SECTIONS IN AND SETS UP PROBABILITIES
λ


λ
NSIGL = NLAST
CALL SETNT(NLAST, NMØST)
STORES NLAST INTERNALLY FOR USE
BY GETNT AND STØRNT;
INCREASES NLAST BY 12*NMØST
IS THIS
A FISSION
PROBLEM
YES
(NFISTP > 0)
NFISBN = NLAST
NLAST = NLAST + 7*NMØST
(FISSION BANK SET UP)
NO
STORE NLAST AND NLEFT
(= NLEFT - NLAST) IN
COMMON USER
WAS LIMIT ON
BLANK COMMON
EXCEEDED?
YES
(NLEFT < 0)
PRINT HUMOROUS
MESSAGE
CALL EXIT
NO
ZERO IG IN THE BANK
CALL SCØRIN
USER ROUTINE WHICH
READS IN NECESSARY
ANALYSIS DATA
μ


B
IS THIS AN ADJOINT PROBLEM?
NO
YES (IADJM > 0)
REVERSE ARRAYS FS, NFS ENERGY, VELOCITY
WAS SOURCE SPECTRUM INPUT?
YES (ISØUR ≤ 0)
PRINT ARRAY FS
PRINT ARRAYS OF ENERGIES AND VELOCITIES
ZERO ARRAYS NDEAD, DEADWT, AND NPSCL
IS THIS A COMBINED NEUTRON AND GAMMA PROBLEM?
YES (NGPQTN*NGPQTG > 0)
STORE VALUES OF GAMMA GENERATION PROBABILITY IN ARRAY BEGINNING IN CELL LØCFSN + 2* NMTG*MEDIA + 1
IS THIS A FISSION PROBLEM?
NO
RETURN
YES NFISTP > 0
IS THIS AN ADJOINT PROBLEM?
NO
STORE VALUES OF FISSION PROBABILITY IN ARRAY BEGINNING IN CELL LØCFSN + 1
CONVERT FISSION SOURCE ENERGY DISTRIBUTIONS TO CDF AND STORE ARRAY BEGINNING IN CELL LØCFSN + NMTG*MEDIA + 1
RETURN
YES IADJM > 0
STORE FISSION SOURCE ENERGY DISTRIBUTION IN FISSION PROBABILITY ARRAY
CONVERT VALUES OF FISSION PROBABILITY TO CDF AND STORE IN FISSION SOURCE ENERGY ARRAY
RETURN

Function IWEEK (MØNTH, IDAT, IYEAR)

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 ≤ 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.

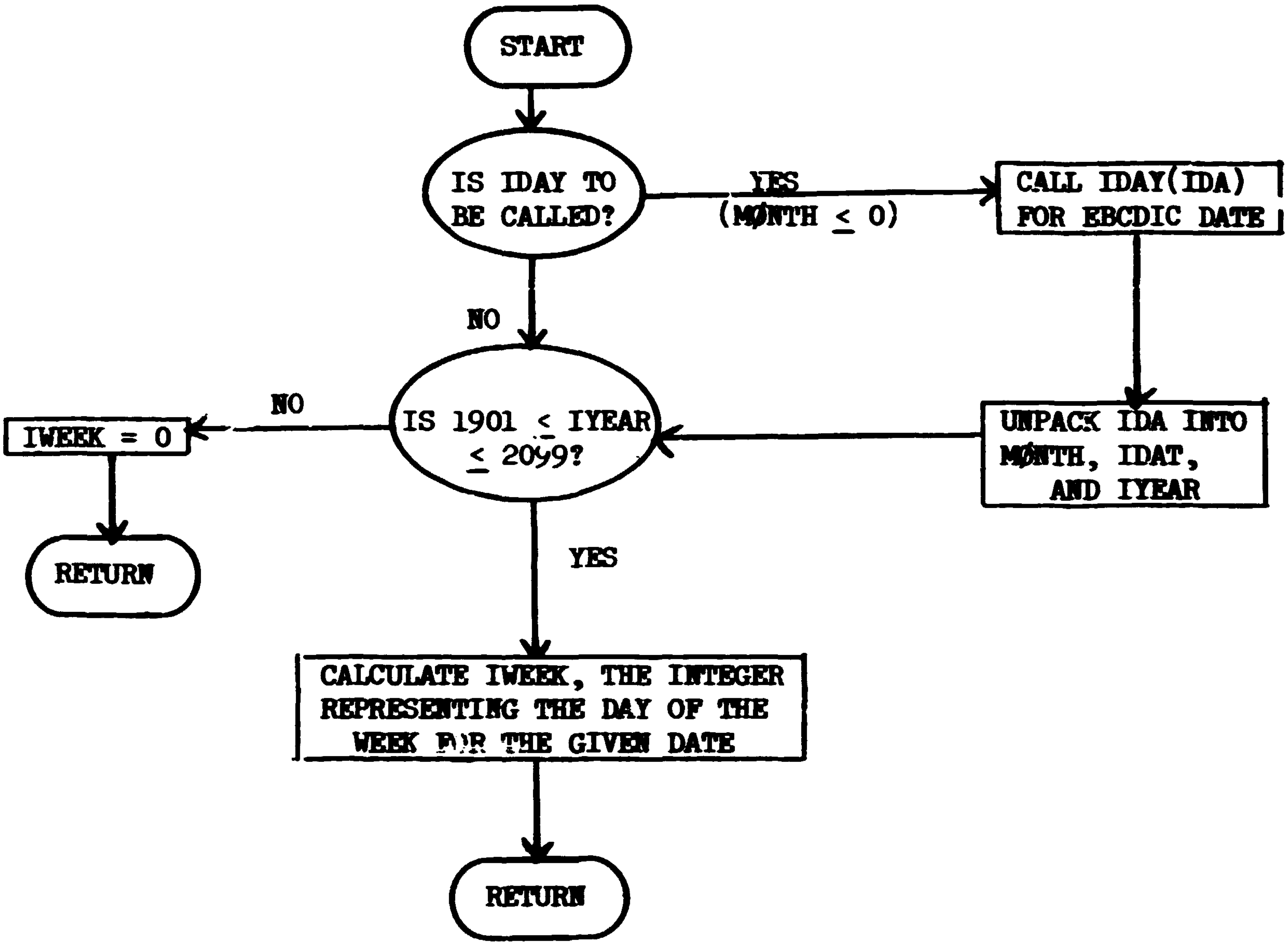
Variables required:

MØNTH ≤ 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.

Function IWEEK (MØNTH,IDAT,IYEAR)


START
IS IDAY TO BE CALLED?
YES (MØNTH ≤ 0)
CALL IDAY(IDA) FOR EBCDIC DATE
NO
UNPACK IDA INTO MØNTH, IDAT, AND IYEAR
IS 1901 ≤ IYEAR ≤ 2099?
NO
IWEEK = 0
RETURN
YES
CALCULATE IWEEK, THE INTEGER REPRESENTING THE DAY OF THE WEEK FOR THE GIVEN DATE
RETURN

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: FSØUR, GETWT, SØURCE, GTISØ, STØRNT, BANKR(1), LØØKZ

Commons required: NUTRØN, FISBNK, APØLLØ, GEØMC

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 subroutine INPUT 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 8

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

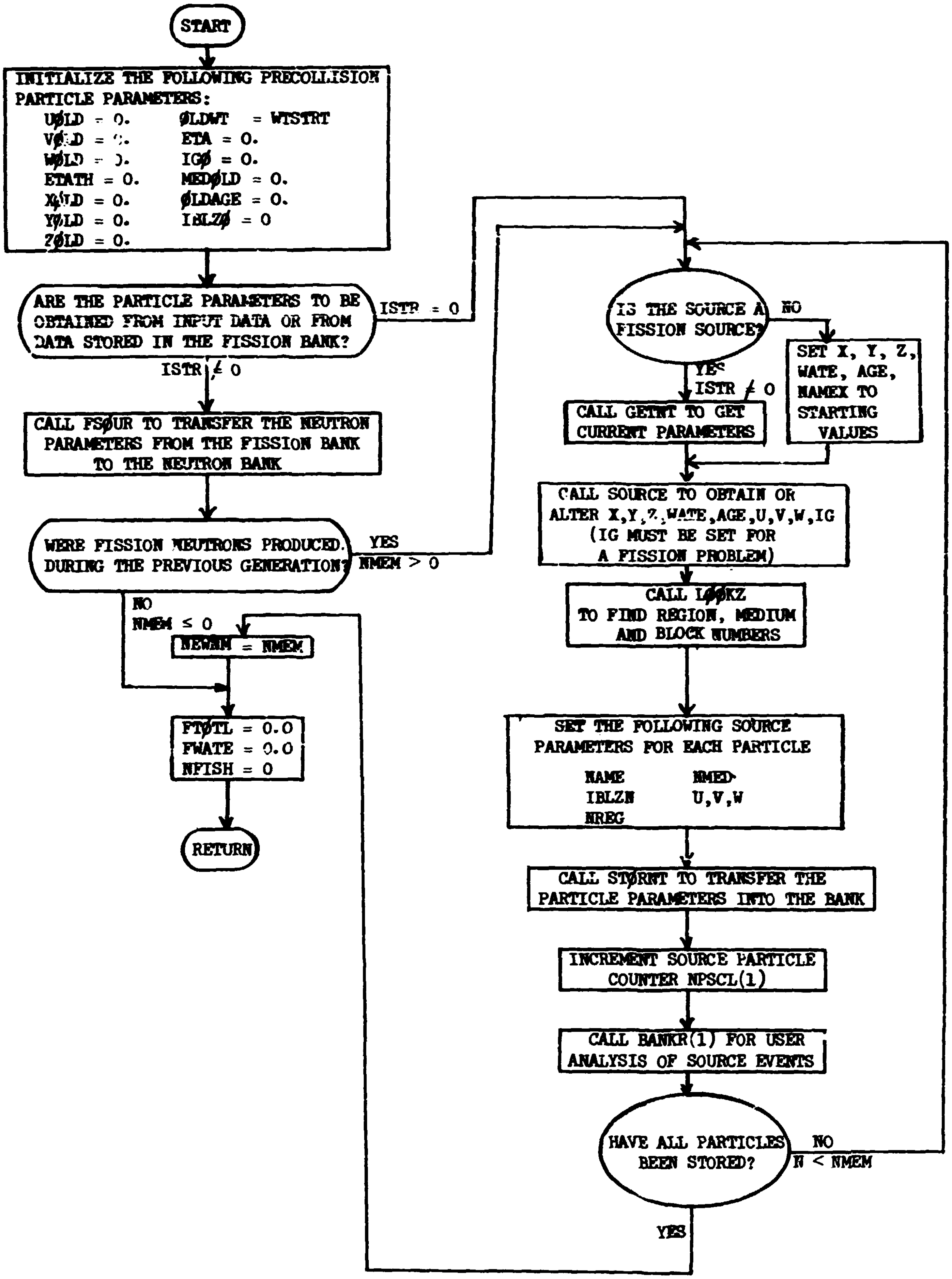
X,Y,Z, WATE, AGE, NAMEX,
IBLZN, NREG, NMED, NAME,
U, V, W, IG } parameters set for each particle generated, put in NUTRØN common, see page 12

NPSCL(1) - counter for number of sources

NEWNM - set to name of last particle generated

FTØTL
FWATE } zeroed for the next batch
NFISH

Subroutine MSØUR


START
INITIALIZE THE FOLLOWING PRECOLLISION PARTICLE PARAMETERS:
UØLD = 0.
VØLD = 0.
WØLD = 0.
ETATH = 0.
XØLD = 0.
YØLD = 0.
ZØLD = 0.
ØLDWT = WTSTRT
ETA = 0.
IGØ = 0.
MEDØLD = 0.
ØLDAGE = 0.
IBLZØ = 0
ARE THE PARTICLE PARAMETERS TO BE OBTAINED FROM INPUT DATA OR FROM DATA STORED IN THE FISSION BANK?
ISTR = 0
ISTR ≠ 0
CALL FSØUR TO TRANSFER THE NEUTRON PARAMETERS FROM THE FISSION BANK TO THE NEUTRON BANK
WERE FISSION NEUTRONS PRODUCED DURING THE PREVIOUS GENERATION?
YES
NMEM > 0
NO
NMEM ≤ 0
NEWNM = NMEM
FTØTL = 0.0
FWATE = 0.0
NFISH = 0
RETURN
IS THE SOURCE A FISSION SOURCE?
NO
SET X, Y, Z, WATE, AGE, NAMEX TO STARTING VALUES
YES
ISTR ≠ 0
CALL GETPT TO GET CURRENT PARAMETERS
CALL SOURCE TO OBTAIN OR ALTER X,Y,Z,WATE,AGE,U,V,W,IG (IG MUST BE SET FOR A FISSION PROBLEM)
CALL LØØKZ TO FIND REGION, MEDIUM AND BLOCK NUMBERS
SET THE FOLLOWING SOURCE PARAMETERS FOR EACH PARTICLE
NAME
IBLZN
NREG
NMED
U,V,W
CALL STØRPT TO TRANSFER THE PARTICLE PARAMETERS INTO THE BANK
INCREMENT SOURCE PARTICLE COUNTER NPSCL(1)
CALL BANKR(1) FOR USER ANALYSIS OF SOURCE EVENTS
HAVE ALL PARTICLES BEEN STORED?
NO
N < NMEM
YES

Subroutine NXTCØL

This subroutine is called by the main program to determine the spatial coordinates, the block and zone number, particle's age, and nonabsorption 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,

BLZNT - a packed word containing the block and 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,

BLZNT - a packed word containing the block and 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.

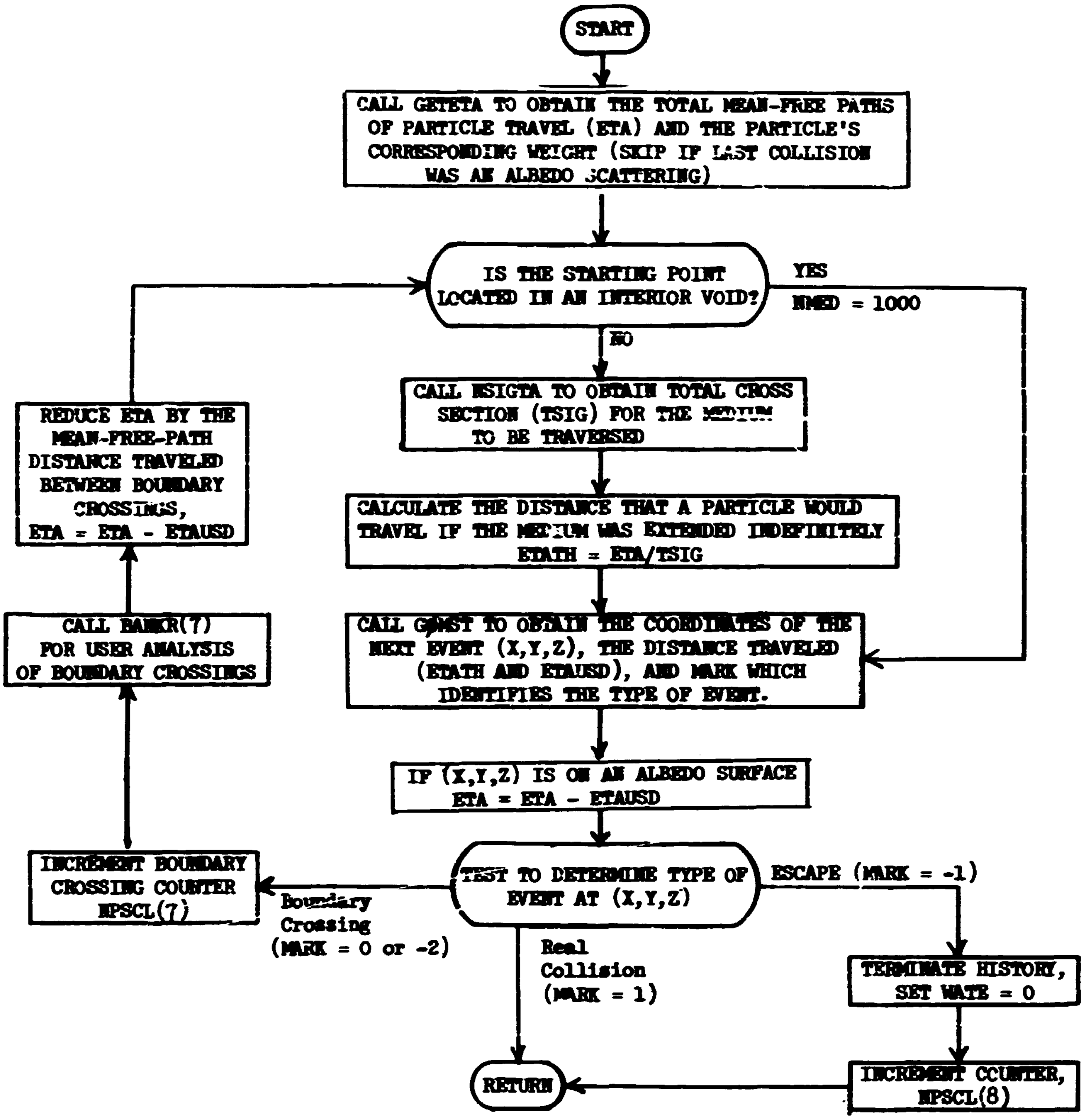
Significant internal variables:

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 NXTCØL


START
CALL GETETA TO OBTAIN THE TOTAL MEAN-FREE PATHS OF PARTICLE TRAVEL (ETA) AND THE PARTICLE'S CORRESPONDING WEIGHT (SKIP IF LAST COLLISION WAS AN ALBEDO SCATTERING)
IS THE STARTING POINT LOCATED IN AN INTERIOR VOID?
YES
NMED = 1000
NO
CALL NSIGTA TO OBTAIN TOTAL CROSS SECTION (TSIG) FOR THE MEDIUM TO BE TRAVERSED
CALCULATE THE DISTANCE THAT A PARTICLE WOULD TRAVEL IF THE MEDIUM WAS EXTENDED INDEFINITELY ETATH = ETA/TSIG
CALL GØMST TO OBTAIN THE COORDINATES OF THE NEXT EVENT (X,Y,Z), THE DISTANCE TRAVELED (ETATH AND ETAUSD), AND MARK WHICH IDENTIFIES THE TYPE OF EVENT.
IF (X,Y,Z) IS ON AN ALBEDO SURFACE ETA = ETA - ETAUSD
TEST TO DETERMINE TYPE OF EVENT AT (X,Y,Z)
ESCAPE (MARK = -1)
Boundary Crossing (MARK = 0 or -2)
Real Collision (MARK = 1)
REDUCE ETA BY THE MEAN-FREE-PATH DISTANCE TRAVELED BETWEEN BOUNDARY CROSSINGS, ETA = ETA - ETAUSD
CALL BANKR(7) FOR USER ANALYSIS OF BOUNDARY CROSSINGS
INCREMENT BOUNDARY CROSSING COUNTER NPSCL(7)
TERMINATE HISTORY, SET WATE = 0
INCREMENT COUNTER, NPSCL(8)
RETURN

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, RNDØUT, GETNT, ØUTPT2

Commons required: Blank, NUTRØN, APØLLØ, FISBNK

Variables required: (nearly all variables from NUTRØN common, see page 12 for definition)

NITS, ITERS, NMEM, RANDØM

NPSCL(I).

NØRMF, NFISH, NKCALC, NSPLT, NKILL (from common APØLLØ, see page 8)

Significant internal variables:

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

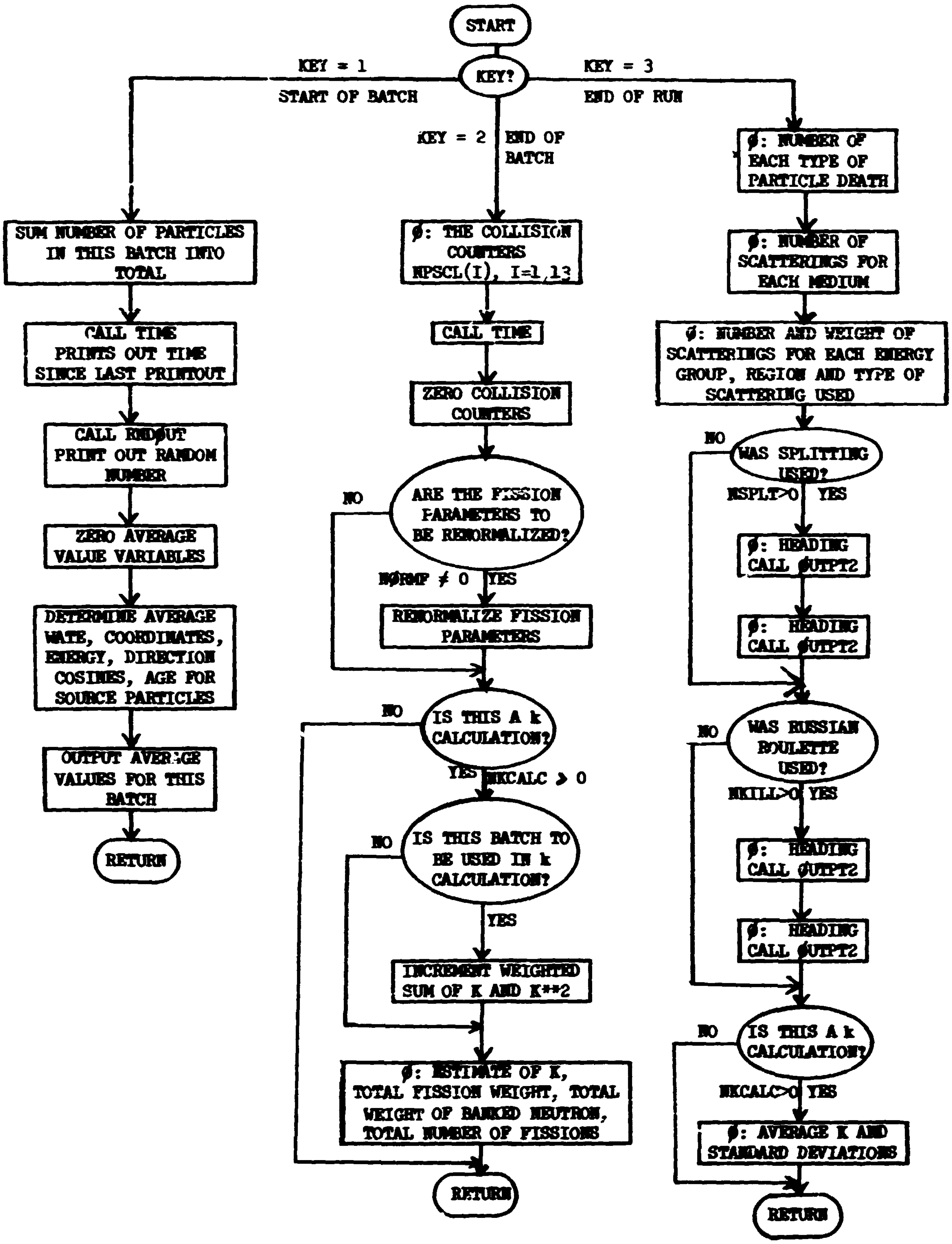
SWATE - the sum of the source particle weights,

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 ØUTPT (KEY)


START
KEY?
KEY = 1
START OF BATCH
KEY = 2 END OF BATCH
KEY = 3
END OF RUN
SUM NUMBER OF PARTICLES IN THIS BATCH INTO TOTAL
CALL TIME PRINTS OUT TIME SINCE LAST PRINTOUT
CALL RNDØUT PRINT OUT RANDOM NUMBER
ZERO AVERAGE VALUE VARIABLES
DETERMINE AVERAGE WATE, COORDINATES, ENERGY, DIRECTION COSINES, AGE FOR SOURCE PARTICLES
OUTPUT AVERAGE VALUES FOR THIS BATCH
RETURN
Ø: THE COLLISION COUNTERS NPSCL(I), I=1,13
CALL TIME
ZERO COLLISION COUNTERS
ARE THE FISSION PARAMETERS TO BE RENORMALIZED?
NO
NØRMF ≠ 0 YES
RENORMALIZE FISSION PARAMETERS
IS THIS A k CALCULATION?
NO
YES NKCALC > 0
IS THIS BATCH TO BE USED IN k CALCULATION?
NO
YES
INCREMENT WEIGHTED SUM OF K AND K**2
Ø: ESTIMATE OF K, TOTAL FISSION WEIGHT, TOTAL WEIGHT OF BANKED NEUTRON, TOTAL NUMBER OF FISSIONS
RETURN
Ø: NUMBER OF EACH TYPE OF PARTICLE DEATH
Ø: NUMBER OF SCATTERINGS FOR EACH MEDIUM
Ø: NUMBER AND WEIGHT OF SCATTERINGS FOR EACH ENERGY GROUP, REGION AND TYPE OF SCATTERING USED
WAS SPLITTING USED?
NO
NSPLT>0 YES
Ø: HEADING CALL ØUTPT2
Ø: HEADING CALL ØUTPT2
WAS RUSSIAN ROULETTE USED?
NO
NKILL>0 YES
Ø: HEADING CALL ØUTPT2
Ø: HEADING CALL ØUTPT2
IS THIS A k CALCULATION?
NO
NKCALC>0 YES
Ø: AVERAGE K AND STANDARD DEVIATIONS
RETURN

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

This sutroutine 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,

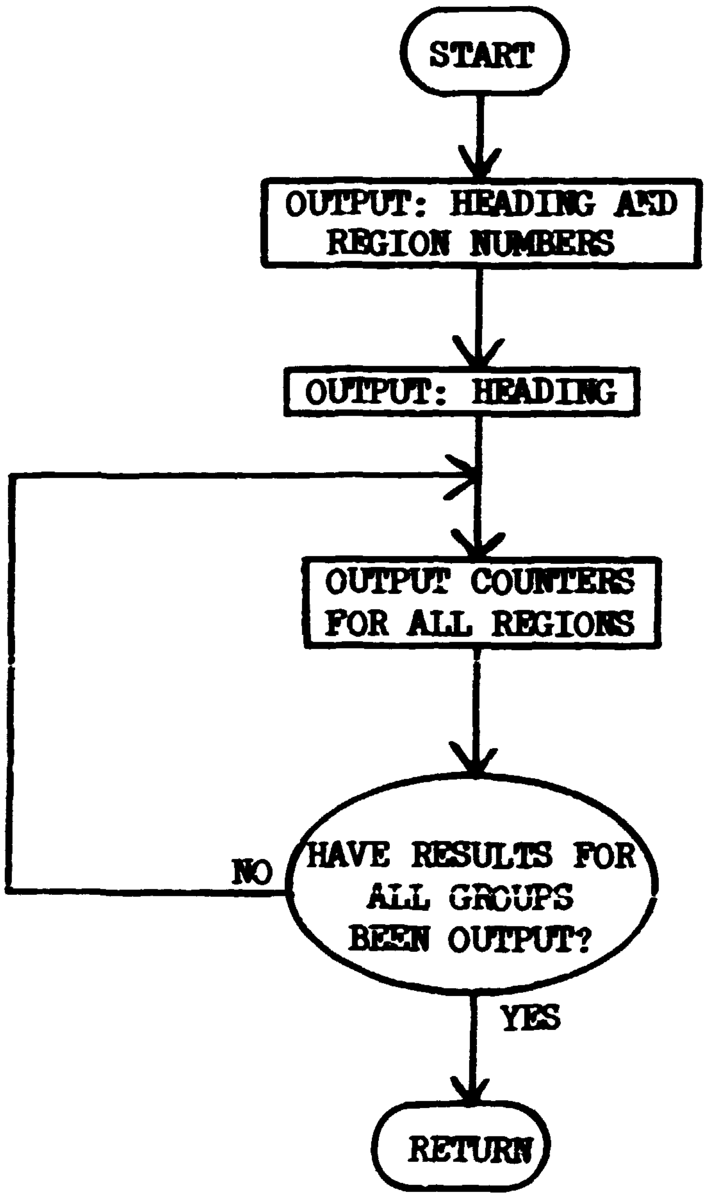
WNI - the two-dimensional array to be output,

MAXGP - the largest group for which Russian roulette and splitting were considered,

MXREG - the number of regions for weight standards,

IØ - logical output tape number.

Subroutine ØUTPT2


START
OUTPUT: HEADING AND REGION NUMBERS
OUTPUT: HEADING
OUTPUT COUNTERS FOR ALL REGIONS
HAVE RESULTS FOR ALL GROUPS BEEN OUTPUT?
NO
YES
RETURN

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 INPUT 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: INPUT

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.

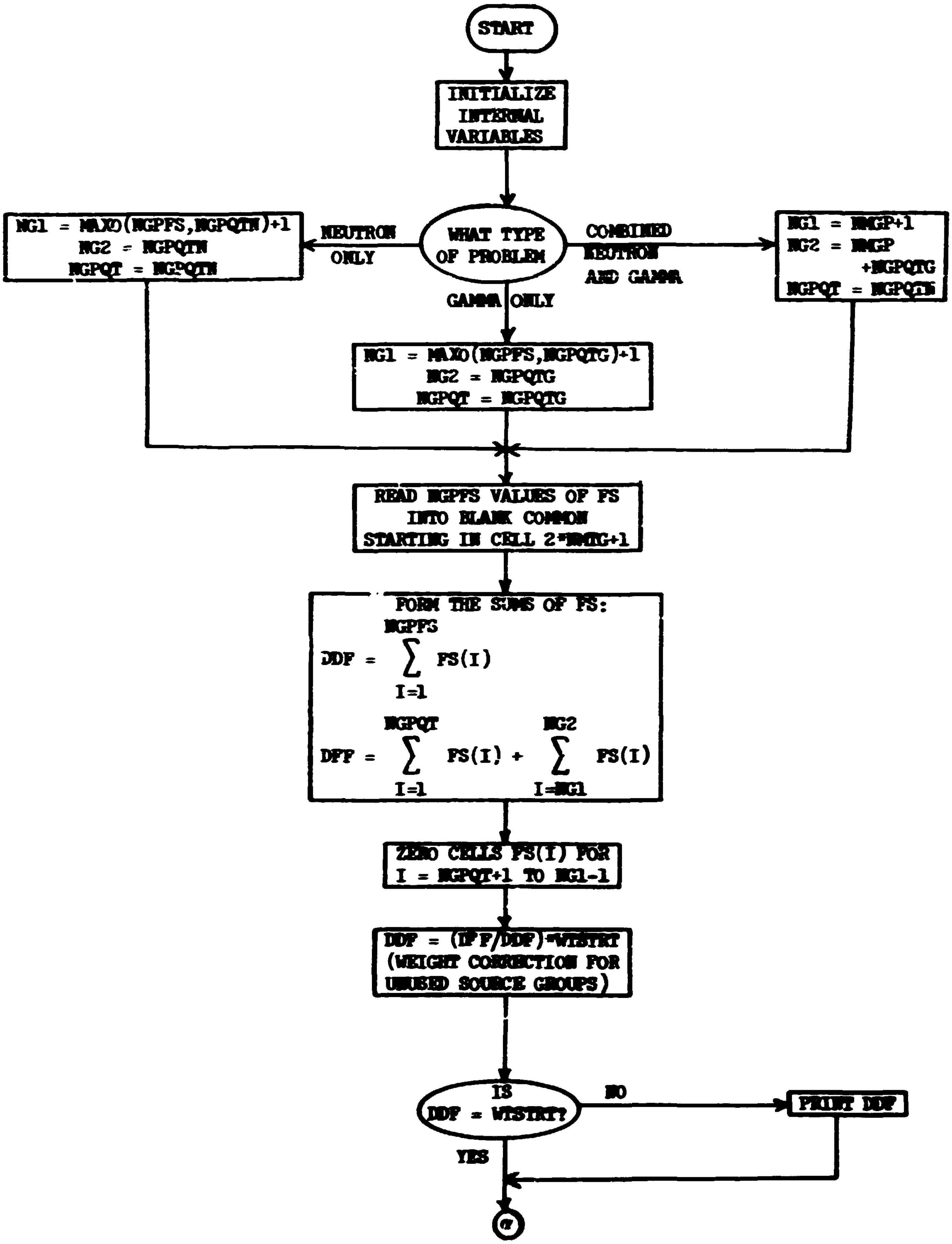
Significant internal variables:

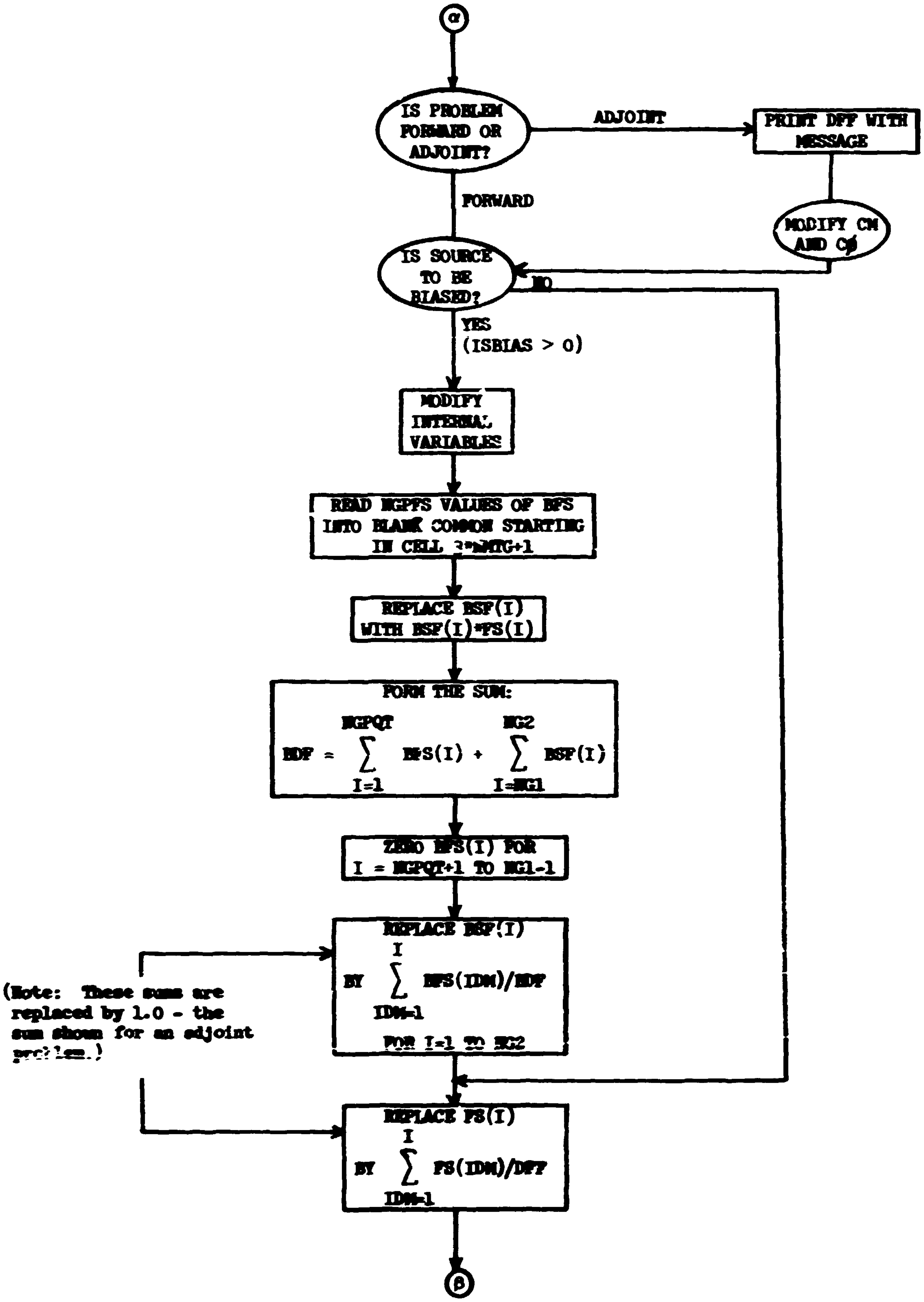
NGPQT - set to NGPQTN if neutron only or combined problem,
set to NGPQTG if gamma only problem,

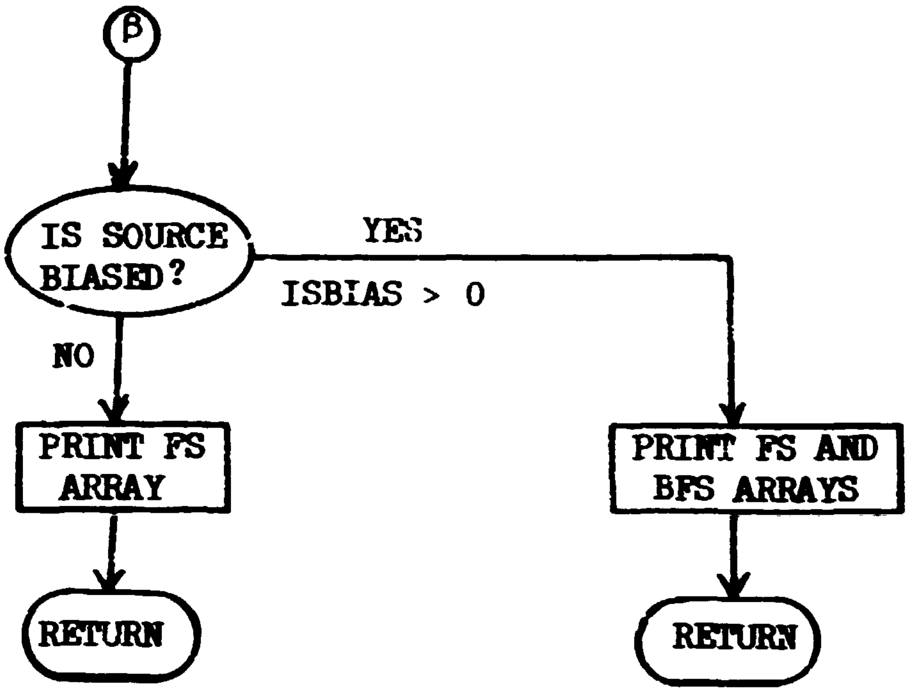
NG1 - set to the largest of NGPFS, NGPQTN, NGPQTG for single particle problem, set to NMGP+1 for combined problem,

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

Subroutine SØRIN (DFF,NGPFS)


START
INITIALIZE INTERNAL VARIABLES
WHAT TYPE OF PROBLEM
NEUTRON ONLY
NG1 = MAXO(NGPFS,NGPQTN)+1
NG2 = NGPQTN
NGPQT = NGPQTN
COMBINED NEUTRON AND GAMMA
NG1 = NMGP+1
NG2 = NMGP +NGPQTG
NGPQT = NGPQTN
GAMMA ONLY
NG1 = MAXO(NGPFS,NGPQTG)+1
NG2 = NGPQTG
NGPQT = NGPQTG
READ NGPFS VALUES OF FS INTO BLANK COMMON STARTING IN CELL 2*NMTG+1
FORM THE SUMS OF FS:
DDF = $\sum_{I=1}^{NGPFS} FS(I)$
DFF = $\sum_{I=1}^{NGPQT} FS(I) + \sum_{I=NG1}^{NG2} FS(I)$
ZERO CELLS FS(I) FOR I = NGPQT+1 TO NG1-1
DDF = (DFF/DDF)*WTSTRT (WEIGHT CORRECTION FOR UNUSED SOURCE GROUPS)
IS DDF = WTSTRT?
NO
PRINT DDF
YES
@


α
IS PROBLEM FORWARD OR ADJOINT?
ADJOINT
PRINT DFF WITH MESSAGE
MODIFY CM AND CØ
FORWARD
IS SOURCE TO BE BIASED?
NO
YES (ISBIAS > 0)
MODIFY INTERNAL VARIABLES
READ NGPFS VALUES OF BFS INTO BLANK COMMON STARTING IN CELL 3*NMTG+1
REPLACE BSF(I) WITH BSF(I)*FS(I)
FORM THE SUM:
NDF = $\sum_{I=1}^{NGPQT}$ BFS(I) + $\sum_{I=NG1}^{NG2}$ BSF(I)
ZERO BFS(I) FOR I = NGPQT+1 TO NG1-1
REPLACE BSF(I) BY $\sum_{IDM=1}^{I}$ BFS(IDM)/NDF FOR I=1 TO NG2
(Note: These sums are replaced by 1.0 - the sum shown for an adjoint problem.)
REPLACE FS(I) BY $\sum_{IDM=1}^{I}$ FS(IDM)/DFF
β


β
IS SOURCE BIASED?
YES
ISBIAS > 0
NO
PRINT FS ARRAY
PRINT FS AND BFS ARRAYS
RETURN
RETURN

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 8

WTHIR - weight standard for splitting,

WTLØR - weight standard for Russian roulette,

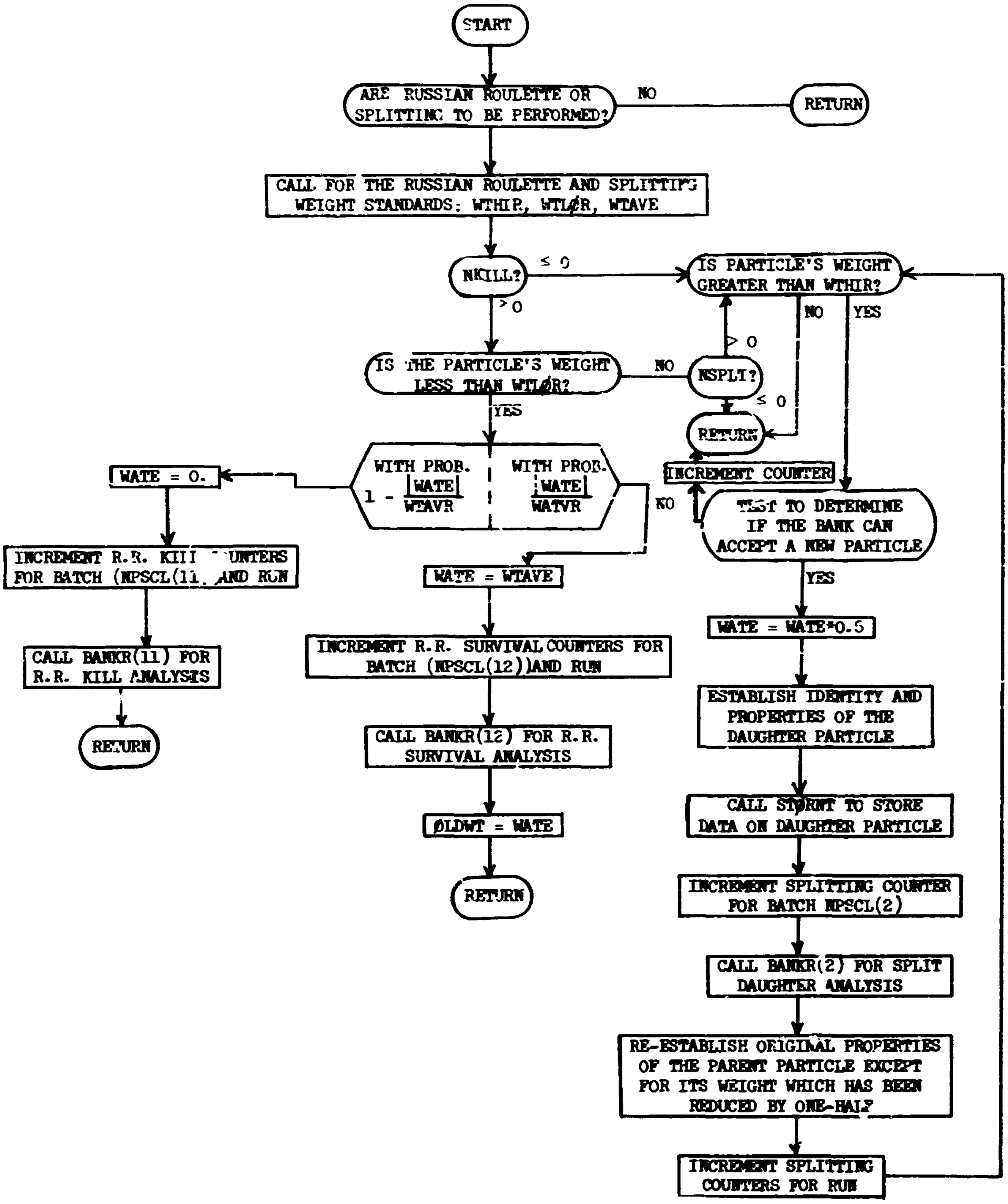
WTAVE - weight assigned to particle which survives Russian roulette.

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.


START
ARE RUSSIAN ROULETTE OR SPLITTING TO BE PERFORMED?
NO
RETURN
CALL FOR THE RUSSIAN ROULETTE AND SPLITTING WEIGHT STANDARDS: WTHIR, WTLØR, WTAVE
NKILL?
≤ 0
> 0
IS PARTICLE'S WEIGHT GREATER THAN WTHIR?
NO
YES
> 0
IS THE PARTICLE'S WEIGHT LESS THAN WTLØR?
NO
NSPLT?
≤ 0
YES
RETURN
WATE = 0.
WITH PROB. 1 - |WATE| / WTAVR
WITH PROB. |WATE| / WATVR
INCREMENT COUNTER
NO
TEST TO DETERMINE IF THE BANK CAN ACCEPT A NEW PARTICLE
INCREMENT R.R. KILL COUNTERS FOR BATCH (NPSCL(11)) AND RUN
WATE = WTAVE
YES
WATE = WATE*0.5
CALL BANKR(11) FOR R.R. KILL ANALYSIS
INCREMENT R.R. SURVIVAL COUNTERS FOR BATCH (NPSCL(12))AND RUN
ESTABLISH IDENTITY AND PROPERTIES OF THE DAUGHTER PARTICLE
RETURN
CALL BANKR(12) FOR R.R. SURVIVAL ANALYSIS
CALL STØRNT TO STORE DATA ON DAUGHTER PARTICLE
ØLDWT = WATE
INCREMENT SPLITTING COUNTER FOR BATCH NPSCL(2)
RETURN
CALL BANKR(2) FOR SPLIT DAUGHTER ANALYSIS
RE-ESTABLISH ORIGINAL PROPERTIES OF THE PARENT PARTICLE EXCEPT FOR ITS WEIGHT WHICH HAS BEEN REDUCED BY ONE-HALF
INCREMENT SPLITTING COUNTERS FOR RUN

Subroutine TIMER (L,A)

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:

```
  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.

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.

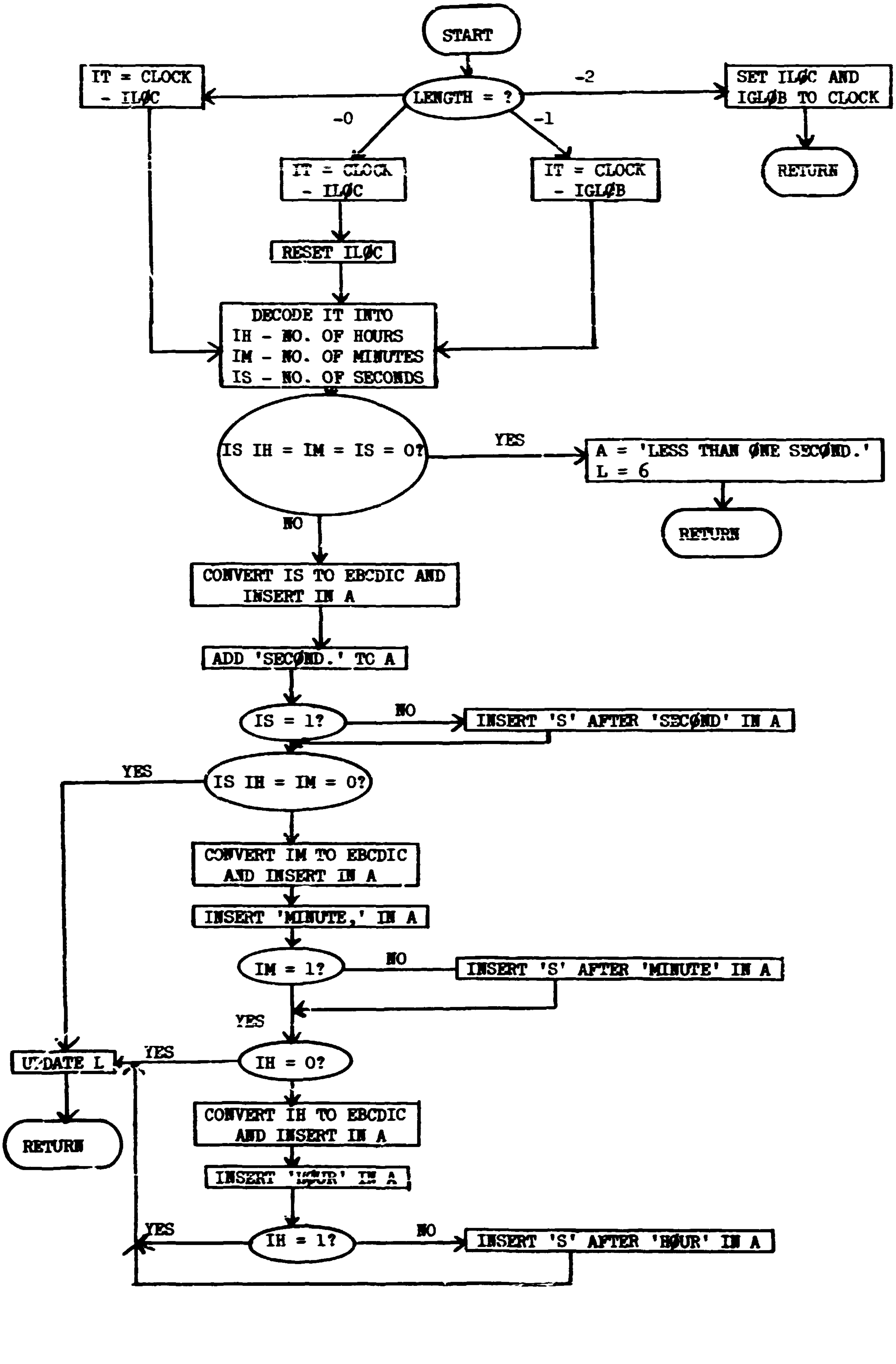
Variables required:

L - see above

Variables modified:

A - see above.

Subroutine TIMER (L,A)



III. Multigroup Cross-Section Module

The function of this module in the multigroup Monte Carlo code is to read ANISN-type 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. Many types of cross sections may be treated, such as neutron only, gamma only, neutron-gamma-ray coupled or gamma rays from a neutron gamma-ray coupled input. Cross sections for either a forward or adjoint solution may be obtained, and the Legendre coefficients for each group-to-group transfer may be retained for next-flight estimation.

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,

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

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 p_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:

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

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. All of the variables used to locate cross sections occur in common LØCSIG. Definitions of the variables which are set up in subroutine XSEC are given in Table V. An outline of the storage of the cross sections in blank common is given in Table VI. 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 Appendix B.

Table V. 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 nonabsorption vector for the first medium
IGABØG	starting location for the gamma-production vector for the first medium
IFPØRG	starting location for the ν* fission probability vector for the first medium
IPNGP	starting location for the primary-secondary transfer probability matrix
IPSPØG	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
ISPØRT*	starting location for temporary storage of downscatter matrix
INPBUF	starting location of input buffer region for the Γ_o table
ISIGØG	starting location for temporary storage of total cross section for element 1
INFPØG*	starting location for temporary stroage 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 by temporary storage

Table V (cont.)

Variable	Definition
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 (see common APØLLØ, page 8)
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
NSCT	number of discrete angles (usually NCØEF/2_{Integral})
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 (see common APØLLØ, see page 8)
NME	indicator for stripping gamma rays from a coupled neutron gamma-ray cross-section set - set equal to number of neutron groups + 1
LØC	same as LØCEPR (see common APØLLØ, see page 8)
INGS	starting location of the indices for starting location of the downscatter vector for each group (primary)
INSG	same as above for secondary
I1, IO	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 cross-section tape if > 0

Table V. (cont.)

Variable	Definition
IDEL	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
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 store Legendre coefficients if > 0
IPUN	switch to print results of bad Legendre coefficients if > 0.

*If Legendre coefficients are to be restored, then:

INPTØG - redefined by JNPUT as starting location of P_1 coefficients for secondary group for medium 1

ITØTSC - redefined by JNPUT as total storage required for all media for each Legendre coefficient

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

Table VI. 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
IPSG	Index to Σ_{GG}(Secondary)	NGG
IDEL	List of Element I.D. Numbers	NELEM*NCOEF if IXTAPE > 0
ISTART	Σ_T	NTG
ISCCØG	Σ_s	NTG
INABØG	Σ_s/Σ_T	NTG
IGABØG	$\Sigma\gamma/\Sigma_T$	NGP
IFPØRG	$\nu\Sigma_f/\Sigma_T$	NTS
IFNGP	$\Sigma_{N\to\gamma}$	NGP*NGG
IFSPØG	$\Sigma^N_{g'\to g}$	$\frac{(NDS+L)(NDS)}{2}$
IDSGØG	$\Sigma^\gamma_{g'\to g}$	$\frac{(NDSG+1)(NDSG)}{2}$
IPRBNG	$P^N_{g'\to g}$	$\frac{(NDS+1)(NDS)}{2}$*NSCT
IPRBGG	$P^\gamma_{g'\to g}$	$\frac{(NDSG+1)(NDSG)}{2}$*NSCT
ISCANG	$A^N_{g'\to g}$	$\frac{(NDS+1)(NDS)}{2}$*NSCT
ISCAGG	$A^\gamma_{g'\to g}$	$\frac{(NDSG+1)(NDSG)}{2}$*NSCT
ISPØRG + ISTART	Repeat for next medium	ISPØRG

Table VI (cont.)

Location*	Information	Size
If ISTAT > 0		
ISPØRT		
	P_1 Coeff. Primary	NDS*NMED
INFPØG		
	P_1 Coeff. Secondary	NDSG*NMED
	Repeat for P_L Coeff.	(NDS + NDSG)*NMED*(NCOEF-2)
NLAST		

*Locations are for index of zero.

Total storage is NTG + NMED*ISPØRG + 3*NMIX + IX + LEG

where:

$$\mathrm{NTG} = \mathrm{NGP} + \mathrm{NGG}$$

$$\mathrm{ISP\O RG} = 4*\mathrm{NTG} + \mathrm{NGP}(1 + \mathrm{NGG}) + \left[\frac{2*\mathrm{NSCT}+1}{2}\right](\mathrm{NDS}+1)(\mathrm{NDS}) + \left[\frac{2*\mathrm{NSCT}+1}{2}\right](\mathrm{NDSG}+1)(\mathrm{NDSG})$$

IX = NELEM*NCOEF if IXTAPE > 0

= 0 otherwise, and

LEG = (NDS+NDSG)*NMED*(NCOEF-1) if ISTAT > 0

= 0 otherwise.

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.

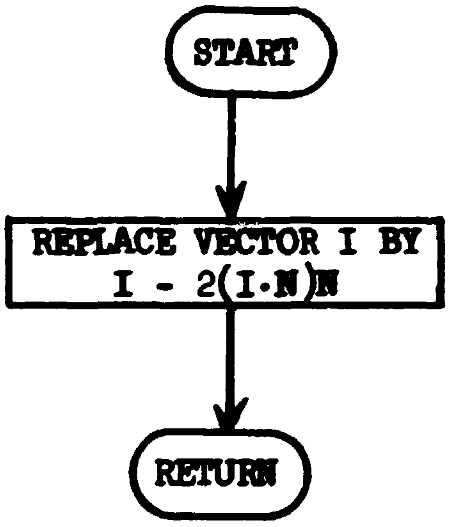
Commons required: NUTRØN, NØRMAL.

Variables required: U, V, W (from common NUTRØN, see page 12)

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

Variables changed: U, V, W.

Subroutine ALBDØ



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.

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: INPUT

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

Commons required: MEAN, RESULT, MØMENT, LØCSIG.

Variables required:

IG1 } indices of group-to-group transfer being
JG1 } calculated

NMØM - number of moments given,

XMØMNT(I) = M_i, i=1, NMØM,

IMØM } > 0 print moments
$\leq$ 0 do not print moments

MX - medium number

IPUN $\leq$ 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,

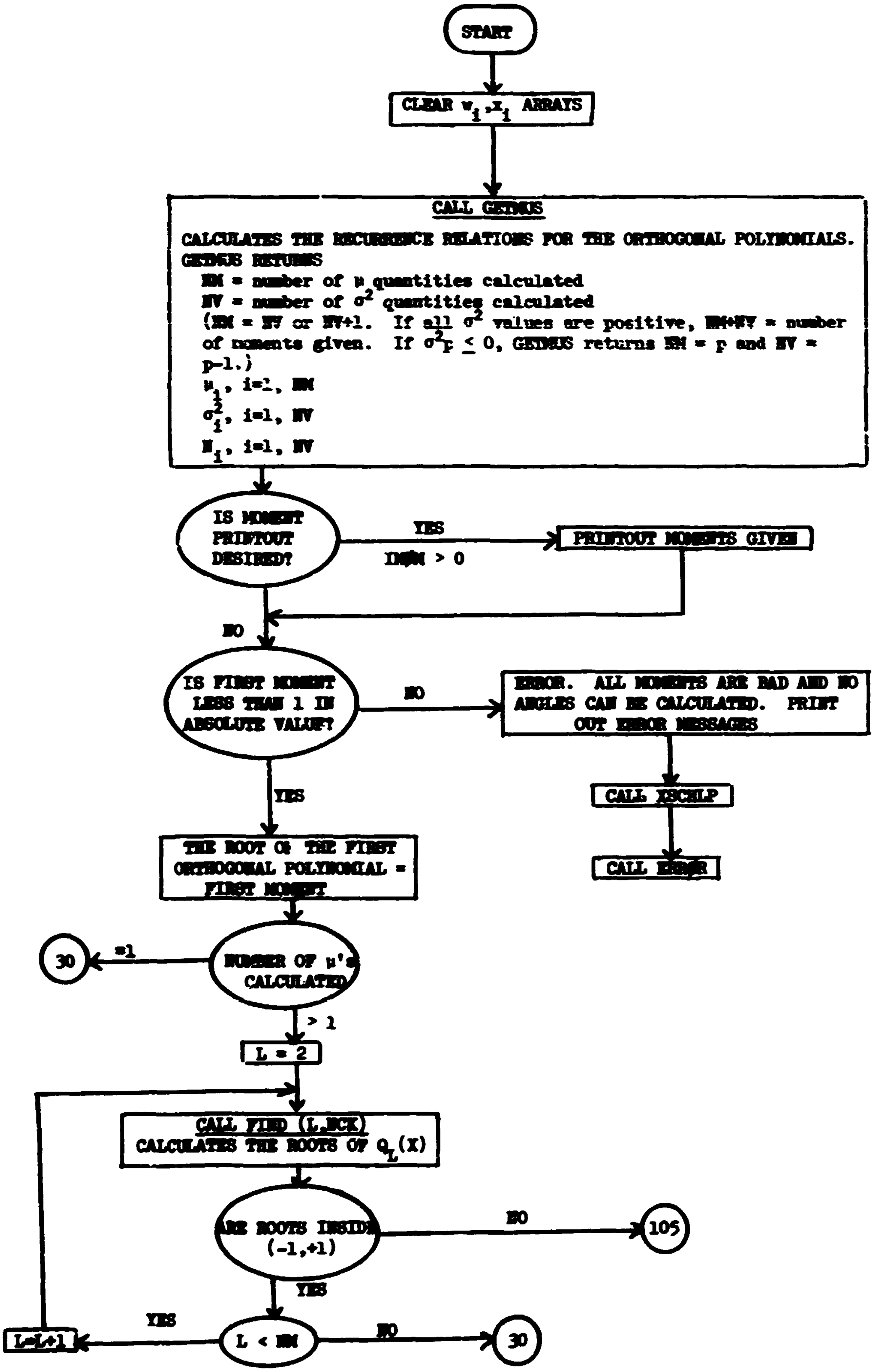
NACC = NM+NV = number of moments accepted.

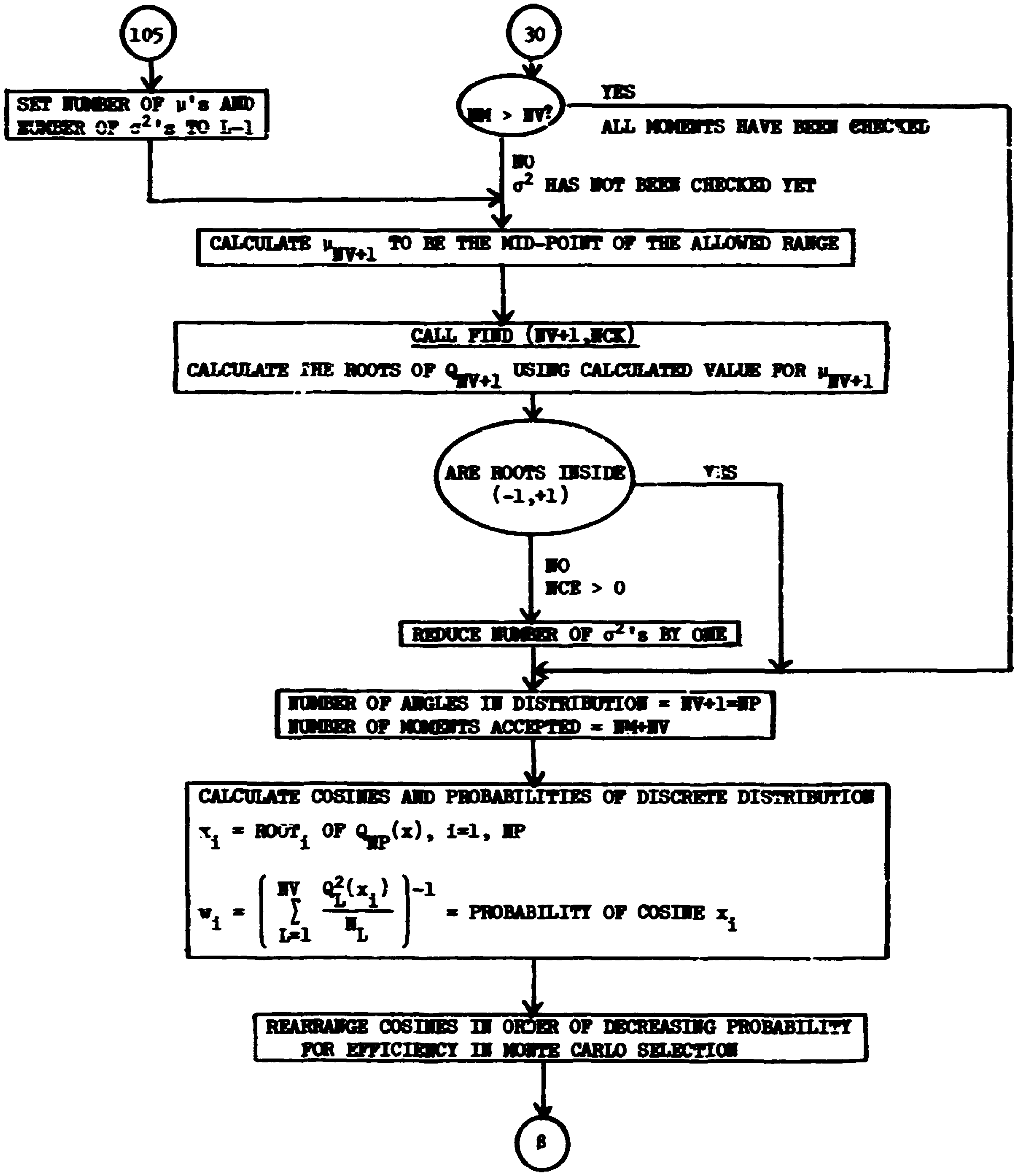
Output:

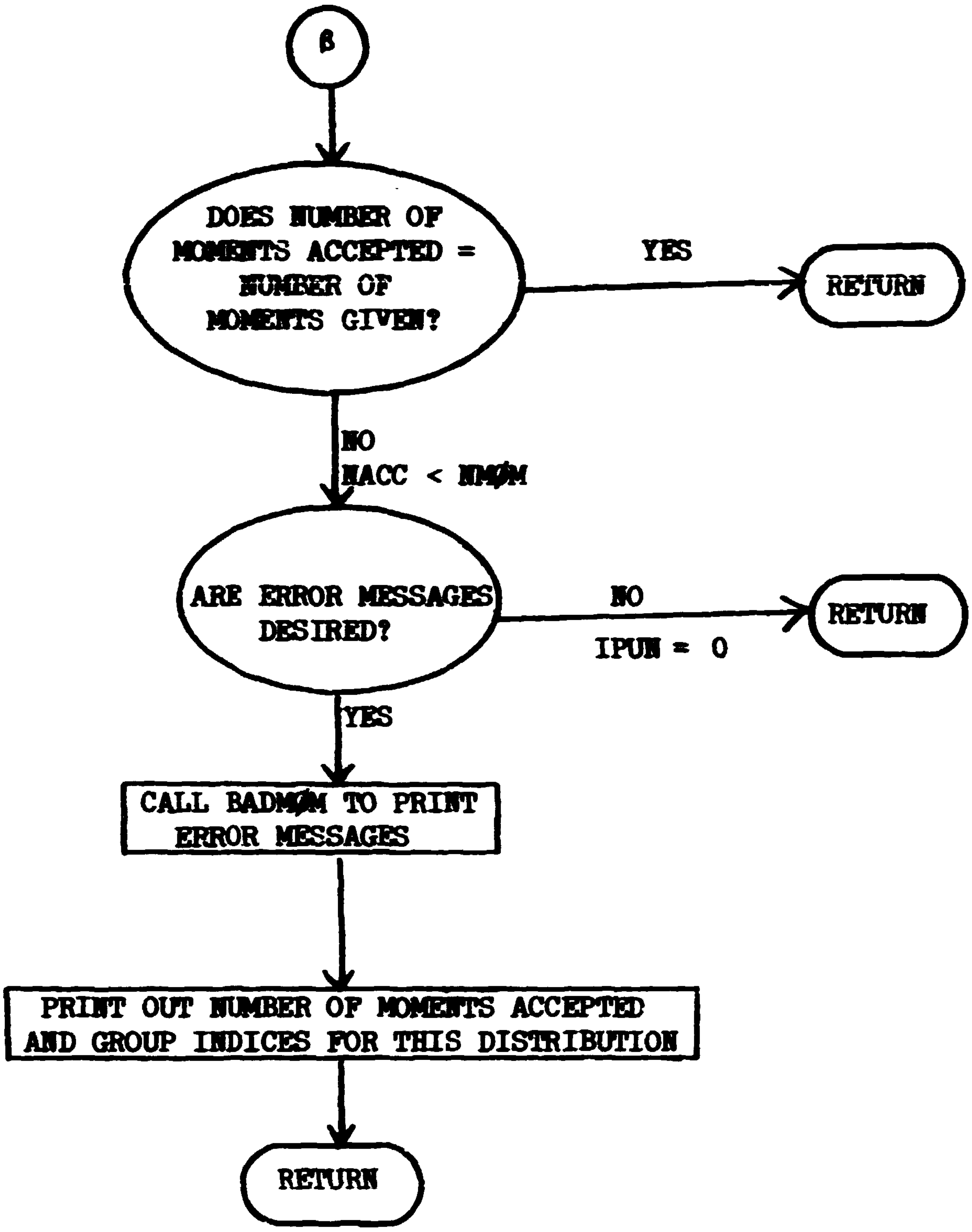
XMØMNT(I) = M_i, I=1, NMØM.

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

Subroutine ANGLES


START
CLEAR w_i, x_i ARRAYS
CALL GETMUS
CALCULATES THE RECURRENCE RELATIONS FOR THE ORTHOGONAL POLYNOMIALS.
GETMUS RETURNS
NM = number of μ quantities calculated
NV = number of σ^2 quantities calculated
(NM = NV or NV+1. If all σ^2 values are positive, NM+NV = number of moments given. If $\sigma^2{}_p \leq 0$, GETMUS returns NM = p and NV = p-1.)
μ_i, i=1, NM
σ_i^2, i=1, NV
N_i, i=1, NV
IS MOMENT PRINTOUT DESIRED?
YES
IDØM > 0
PRINTOUT MOMENTS GIVEN
NO
IS FIRST MOMENT LESS THAN 1 IN ABSOLUTE VALUE?
NO
ERROR. ALL MOMENTS ARE BAD AND NO ANGLES CAN BE CALCULATED. PRINT OUT ERROR MESSAGES
CALL XSCHLP
CALL ERRØR
YES
THE ROOT OF THE FIRST ORTHOGONAL POLYNOMIAL = FIRST MOMENT
30
=1
NUMBER OF μ's CALCULATED
> 1
L = 2
CALL FIND (L,NCK)
CALCULATES THE ROOTS OF $Q_L(X)$
ARE ROOTS INSIDE (-1,+1)
NO
105
YES
L=L+1
YES
L < NM
NO
30


105
30
SET NUMBER OF μ's AND NUMBER OF σ²'s TO L-1
NM > NV?
YES
ALL MOMENTS HAVE BEEN CHECKED
NO
σ² HAS NOT BEEN CHECKED YET
CALCULATE μ_NV+1 TO BE THE MID-POINT OF THE ALLOWED RANGE
CALL FIND (NV+1,NCK)
CALCULATE THE ROOTS OF Q_NV+1 USING CALCULATED VALUE FOR μ_NV+1
ARE ROOTS INSIDE (-1,+1)
YES
NO
NCE > 0
REDUCE NUMBER OF σ²'s BY ONE
NUMBER OF ANGLES IN DISTRIBUTION = NV+1=NP
NUMBER OF MOMENTS ACCEPTED = NM+NV
CALCULATE COSINES AND PROBABILITIES OF DISCRETE DISTRIBUTION
x_i = ROOT_i OF Q_NP(x), i=1, NP
w_i = (Σ_{L=1}^{NV} Q²_L(x_i) / N_L)^-1 = PROBABILITY OF COSINE x_i
REARRANGE COSINES IN ORDER OF DECREASING PROBABILITY FOR EFFICIENCY IN MONTE CARLO SELECTION
8


β
DOES NUMBER OF MOMENTS ACCEPTED = NUMBER OF MOMENTS GIVEN?
YES
RETURN
NO
NACC < NMØM
ARE ERROR MESSAGES DESIRED?
NO
IPUN = 0
RETURN
YES
CALL BADMØM TO PRINT ERROR MESSAGES
PRINT OUT NUMBER OF MOMENTS ACCEPTED AND GROUP INDICES FOR THIS DISTRIBUTION
RETURN

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 that 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 - $(\sigma^2)^{min}$,

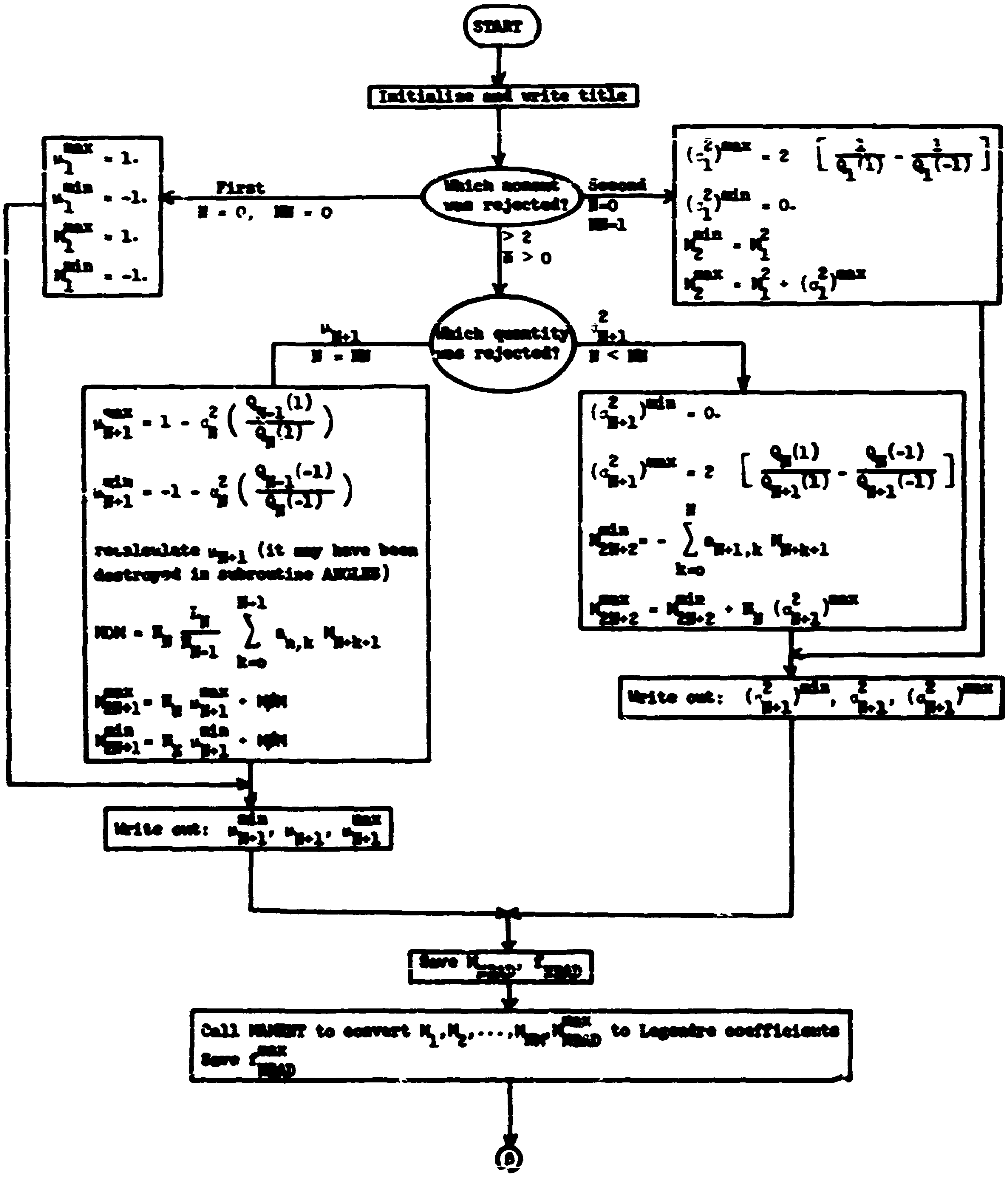
MØMT - M^{max},

MØMB - M^{min},

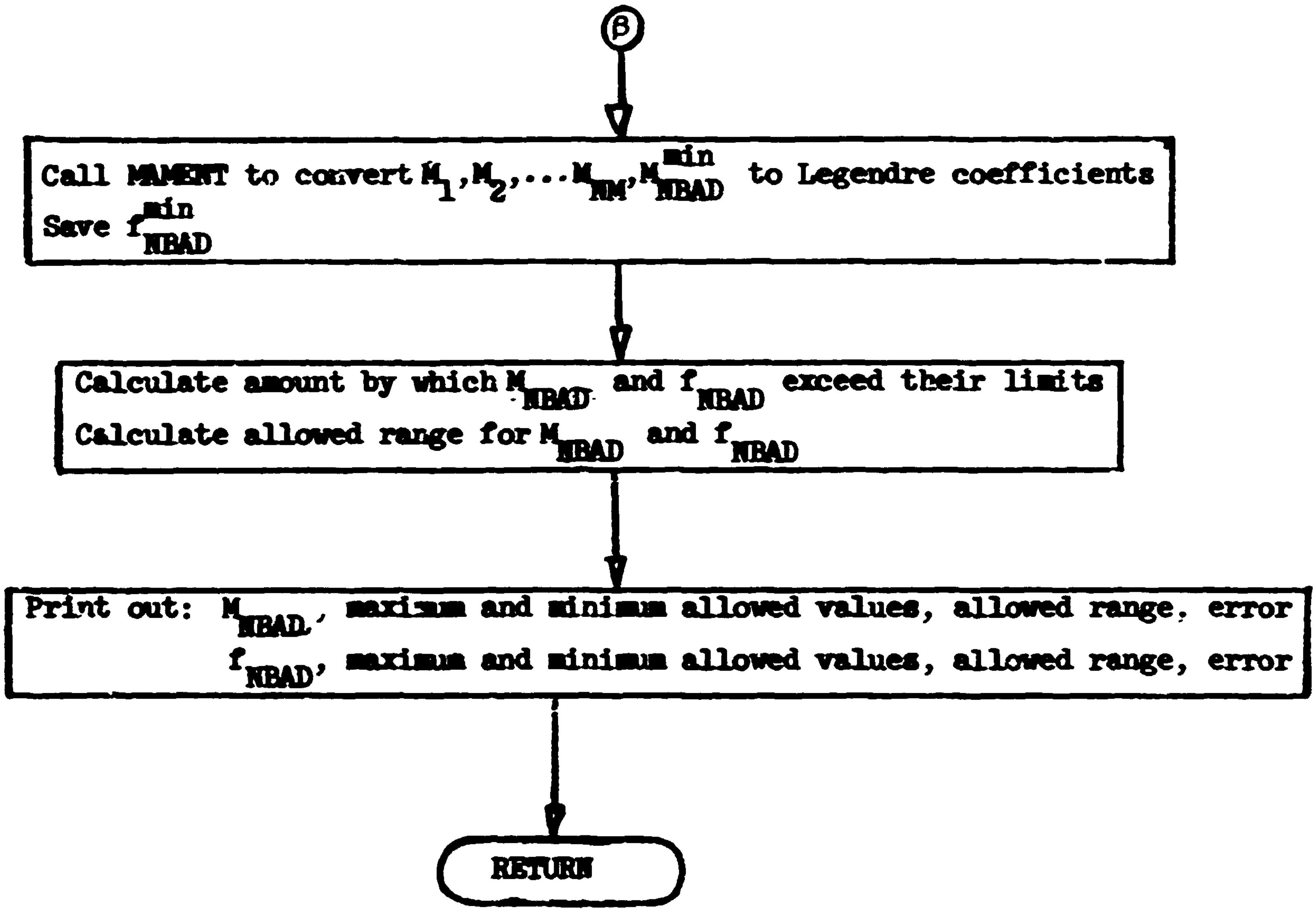
FT - f^{max},

FB - f^{min}.

Subroutine BADMOM



START
Initialise and write title
Which moment was rejected?
First
N = 0, NN = 0
$\mu_1^{max} = 1.$
$\mu_1^{min} = -1.$
$M_1^{max} = 1.$
$M_1^{min} = -1.$
Second
N=0
NN=1
$(\sigma_1^2)^{max} = 2 \left[\frac{1}{Q_1(1)} - \frac{1}{Q_1(-1)} \right]$
$(\sigma_1^2)^{min} = 0.$
$M_2^{min} = M_1^2$
$M_2^{max} = M_1^2 + (\sigma_1^2)^{max}$
> 2
N > 0
Which quantity was rejected?
μ_{N+1}
N = NN
$\mu_{N+1}^{max} = 1 - \sigma_N^2 \left(\frac{Q_{N-1}(1)}{Q_N(1)} \right)$
$\mu_{N+1}^{min} = -1 - \sigma_N^2 \left(\frac{Q_{N-1}(-1)}{Q_N(-1)} \right)$
recalculate μ_{N+1} (it may have been destroyed in subroutine ANGLES)
$MOM = K_N \frac{L_N}{K_{N-1}} \sum_{k=0}^{N-1} a_{N,k} M_{N+k+1}$
$M_{2N+1}^{max} = K_N \mu_{N+1}^{max} + MOM$
$M_{2N+1}^{min} = K_N \mu_{N+1}^{min} + MOM$
σ_{N+1}^2
N < NN
$(\sigma_{N+1}^2)^{min} = 0.$
$(\sigma_{N+1}^2)^{max} = 2 \left[\frac{Q_N(1)}{Q_{N+1}(1)} - \frac{Q_N(-1)}{Q_{N+1}(-1)} \right]$
$M_{2N+2}^{min} = - \sum_{k=0}^{N} a_{N+1,k} M_{N+k+1}$
$M_{2N+2}^{max} = M_{2N+2}^{min} + K_N (\sigma_{N+1}^2)^{max}$
Write out: $\mu_{N+1}^{min}, \mu_{N+1}, \mu_{N+1}^{max}$
Write out: $(\sigma_{N+1}^2)^{min}, \sigma_{N+1}^2, (\sigma_{N+1}^2)^{max}$
Save M_{NBAD}, ξ_{NBAD}
Call MAMENT to convert $M_1, M_2, \ldots, M_{NN}, M_{NBAD}^{max}$ to Legendre coefficients
Save ξ_{NBAD}^{max}
B


β
Call MAMENT to convert $M_1, M_2, \ldots M_{NM}, M^{min}_{NBAD}$ to Legendre coefficients
Save f^{min}_{NBAD}
Calculate amount by which M_{NBAD} and f_{NBAD} exceed their limits
Calculate allowed range for M_{NBAD} and f_{NBAD}
Print out: M_{NBAD}, maximum and minimum allowed values, allowed range, error
f_{NBAD}, maximum and minimum allowed values, allowed range, error
RETURN

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. 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 88)

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,

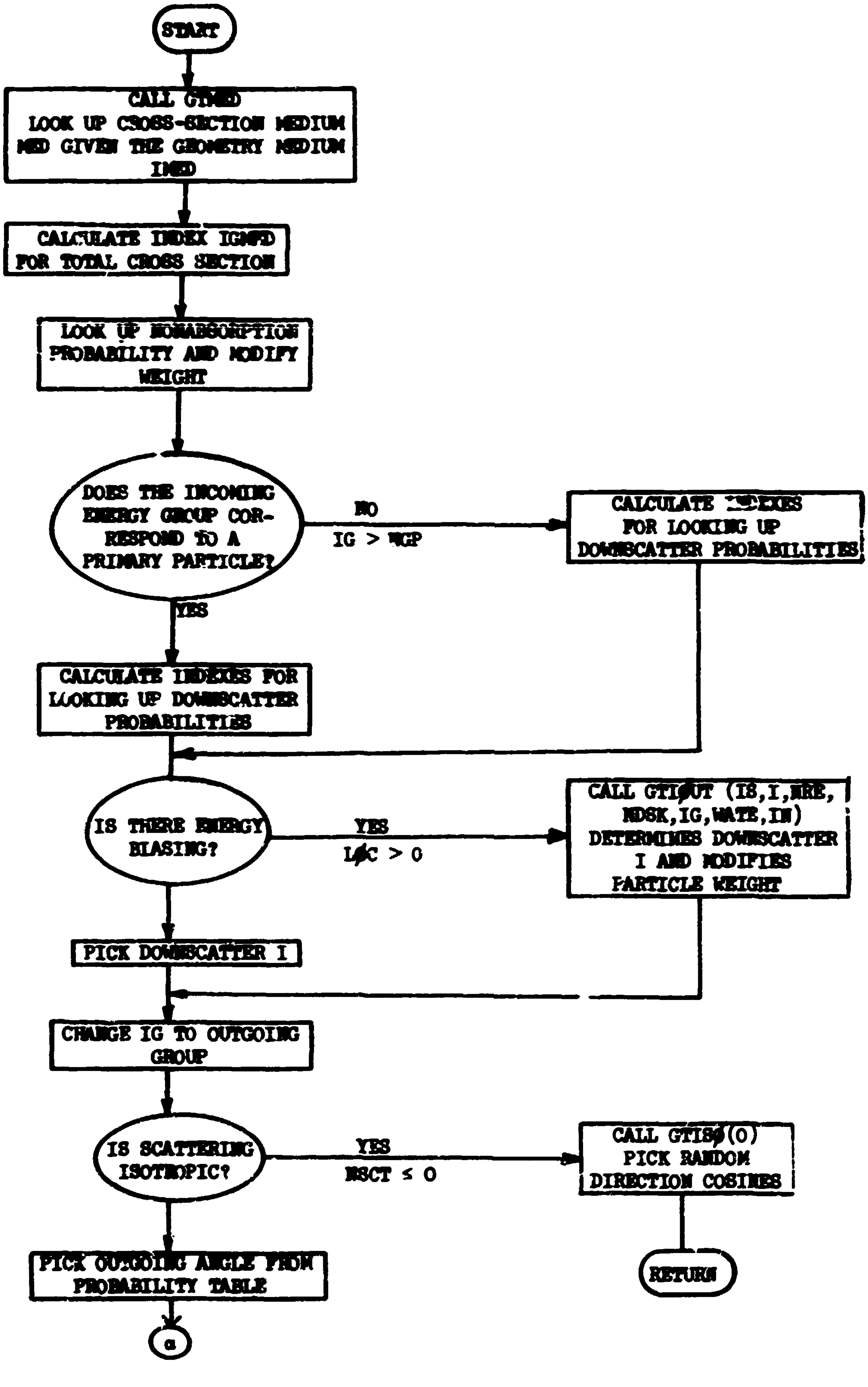
NDSK - number of downscatter groups,

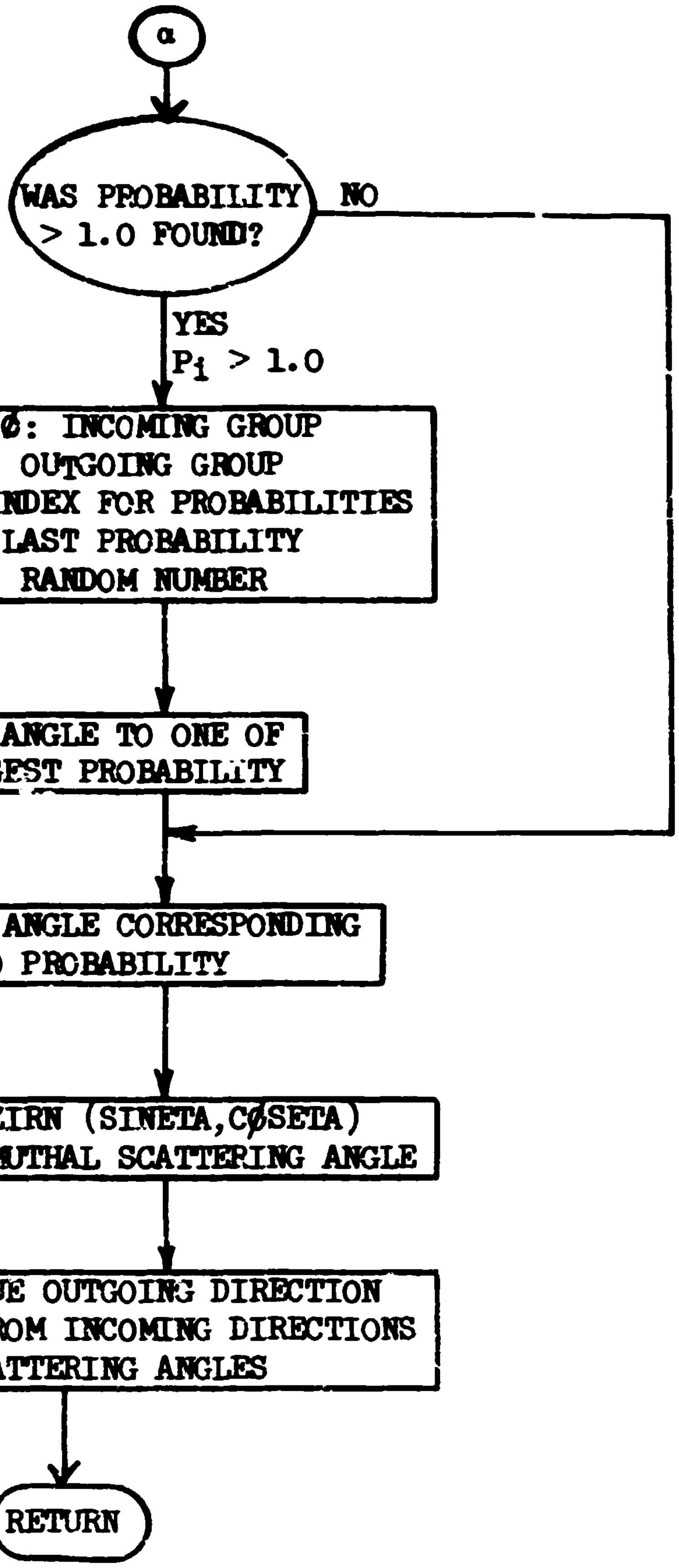
FM - cosine of polar angle of scattering,

SINETA } sine and cosine of azimuthal angle of scattering
CØSETA }

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

Subroutine CØLISN


START
CALL GTMED
LOOK UP CROSS-SECTION MEDIUM MED GIVEN THE GEOMETRY MEDIUM IMED
CALCULATE INDEX IGMED FOR TOTAL CROSS SECTION
LOOK UP NONABSORPTION PROBABILITY AND MODIFY WEIGHT
DOES THE INCOMING ENERGY GROUP CORRESPOND TO A PRIMARY PARTICLE?
NO
IG > NGP
CALCULATE INDEXES FOR LOOKING UP DOWNSCATTER PROBABILITIES
YES
CALCULATE INDEXES FOR LOOKING UP DOWNSCATTER PROBABILITIES
IS THERE ENERGY BIASING?
YES
LØC > 0
CALL GTIØUT (IS,I,NRE, NDSK,IG,WATE,IN) DETERMINES DOWNSCATTER I AND MODIFIES PARTICLE WEIGHT
PICK DOWNSCATTER I
CHANGE IG TO OUTGOING GROUP
IS SCATTERING ISOTROPIC?
YES
NSCT ≤ 0
CALL GTISØ(0) PICK RANDOM DIRECTION COSINES
RETURN
PICK OUTGOING ANGLE FROM PROBABILITY TABLE
a


α
WAS PROBABILITY > 1.0 FOUND?
NO
YES
$P_i > 1.0$
Ø: INCOMING GROUP
OUTGOING GROUP
LAST INDEX FOR PROBABILITIES
LAST PROBABILITY
RANDOM NUMBER
SET ANGLE TO ONE OF LARGEST PROBABILITY
PICK UP ANGLE CORRESPONDING TO PROBABILITY
CALL AZIRN (SINETA,CØSETA)
SELECT AZIMUTHAL SCATTERING ANGLE
DETERMINE OUTGOING DIRECTION COSINES FROM INCOMING DIRECTIONS AND SCATTERING ANGLES
RETURN

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, LØCSIG

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:

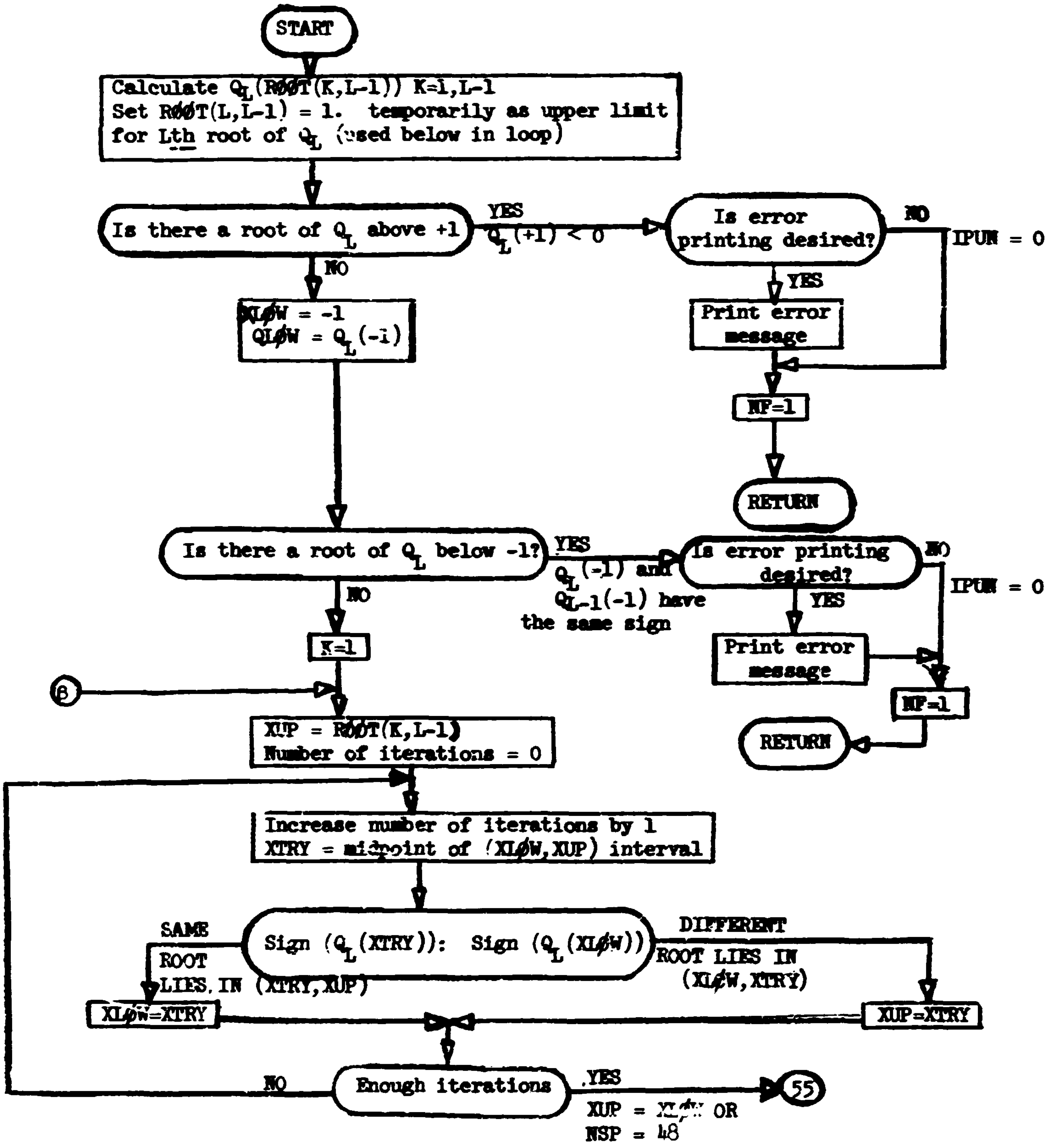
VALUE(K) - Q_L(RØØT(K,L-1)), K = 1,L-1,

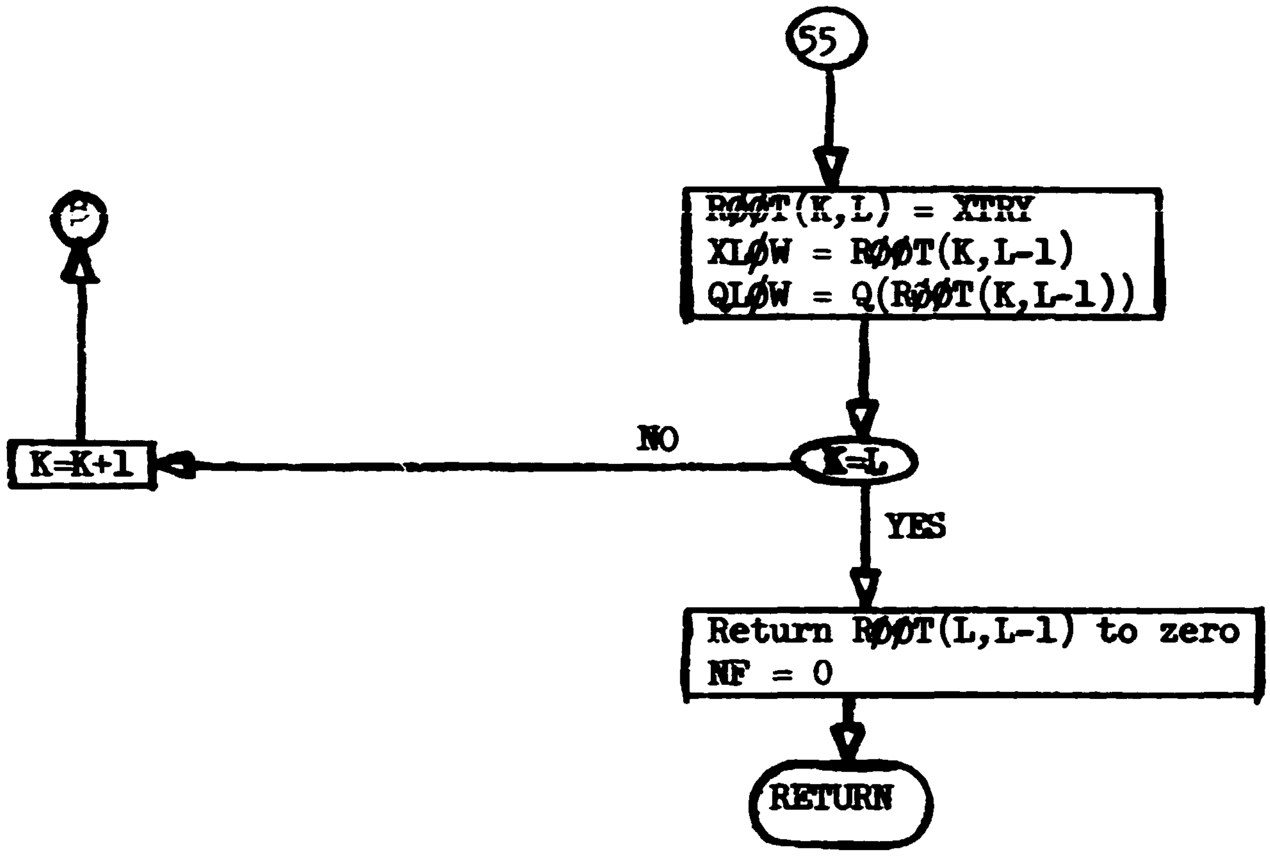
LM1 = L - 1,

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

Limitations: $L \leq 14$.

Subroutine FIND


START
Calculate Q_L(RØØT(K,L-1)) K=1,L-1
Set RØØT(L,L-1) = 1. temporarily as upper limit
for Lth root of Q_L (used below in loop)
Is there a root of Q_L above +1
YES
Q_L(+1) < 0
Is error printing desired?
NO
IPUN = 0
YES
Print error message
NF=1
RETURN
NO
XLØW = -1
QLØW = Q_L(-1)
Is there a root of Q_L below -1?
YES
Q_L(-1) and Q_{L-1}(-1) have the same sign
Is error printing desired?
NO
IPUN = 0
YES
Print error message
NF=1
RETURN
NO
K=1
6
XUP = RØØT(K,L-1)
Number of iterations = 0
Increase number of iterations by 1
XTRY = midpoint of (XLØW,XUP) interval
SAME
ROOT LIES IN (XTRY,XUP)
Sign (Q_L(XTRY)): Sign (Q_L(XLØW))
DIFFERENT
ROOT LIES IN (XLØW,XTRY)
XLØW=XTRY
XUP=XTRY
NO
Enough iterations
YES
XUP = XLØW OR
NSP = 48
55


55
RØØT(K,L) = XTRY
XLØW = RØØT(K,L-1)
QLØW = Q(RØØT(K,L-1))
K=L
NO
K=K+1
YES
Return RØØT(L,L-1) to zero
NF = 0
RETURN

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: INPUT.

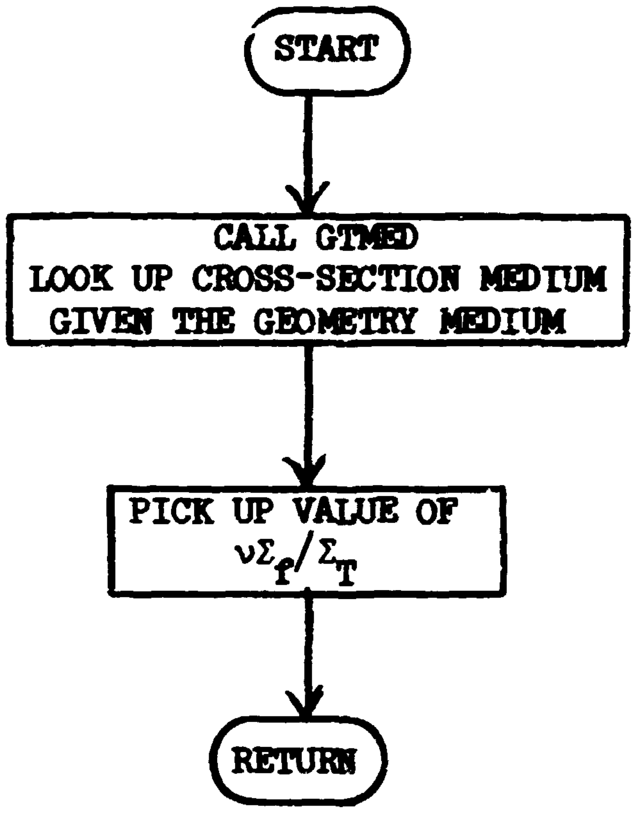
Subroutines called: GTMED

Commons required: Blank, LØCSIG.

Variables required: ISPØRG, IFPØRG IG, MED } (from common LØCSIG, page 89)

Variables changed: PNUF.

Subroutine FISGEN



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.

Called from: GPRØB

Subroutines called: GTMED

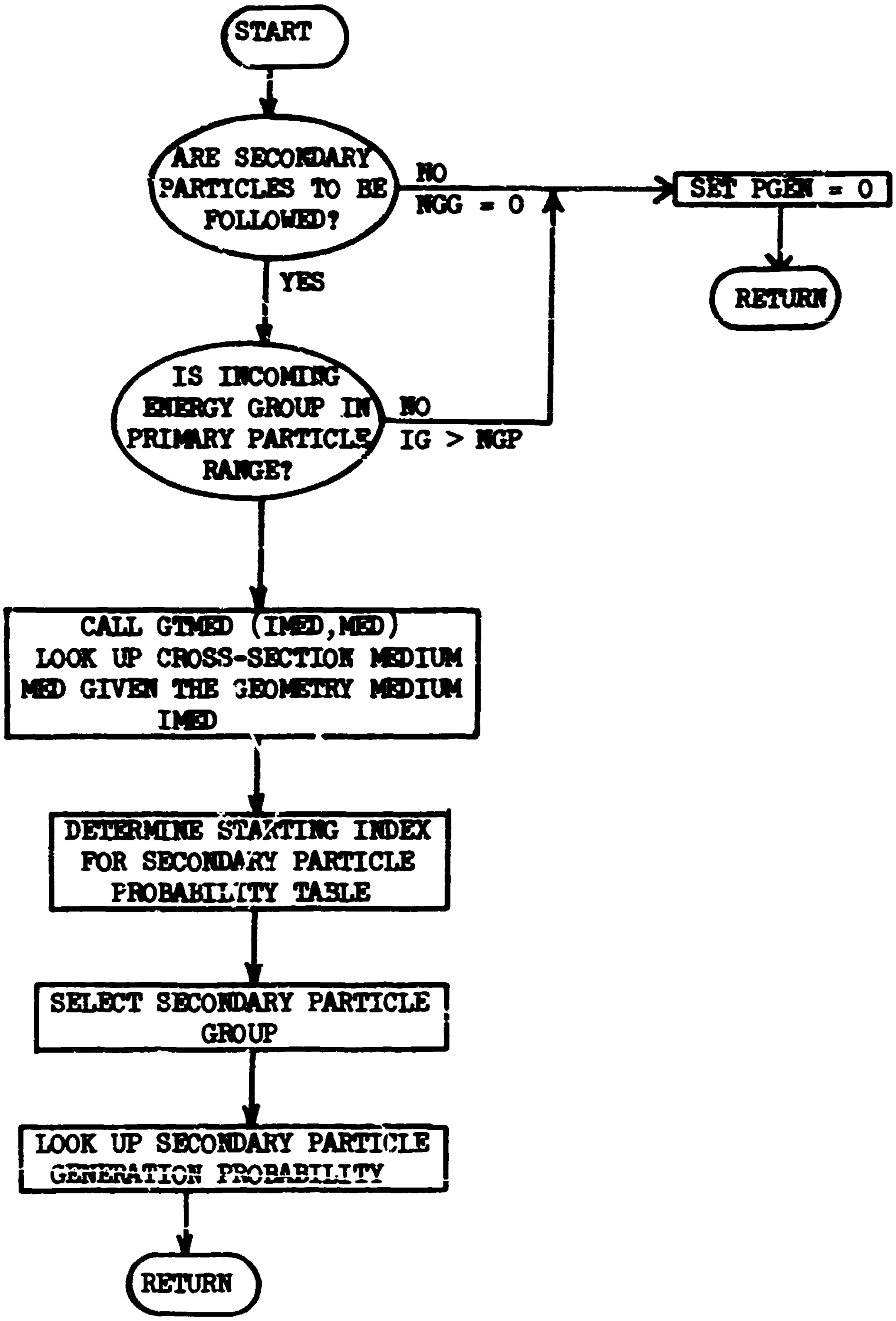
Functions used: FLTRNF

Commons required: Blank, LØCSIG

Variables required: ISPØRG, IFNGP, NGG, IGSBØG (from common LØCSIG, see page 88)
IG - incoming energy group,
IMED - medium of collision site as provided by the geometry module.

Variables changed: PGEN, IGG.

Subroutine GAMGEN


START
ARE SECONDARY PARTICLES TO BE FOLLOWED?
NO
NGG = 0
SET PGEN = 0
RETURN
YES
IS INCOMING ENERGY GROUP IN PRIMARY PARTICLE RANGE?
NO
IG > NGP
CALL GTMED (IMED,MED)
LOOK UP CROSS-SECTION MEDIUM MED GIVEN THE GEOMETRY MEDIUM IMED
DETERMINE STARTING INDEX FOR SECONDARY PARTICLE PROBABILITY TABLE
SELECT SECONDARY PARTICLE GROUP
LOOK UP SECONDARY PARTICLE GENERATION PROBABILITY
RETURN

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

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

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

The calculation proceeds as follows:

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

<u>Step 2</u>: set $i = 3$.

<u>Step 3</u>: calculate L_i from moments and coefficients for $i - 1$.

<u>Step 4</u>: calculate q_i from L_i and N_{i-1}.

<u>Step 5</u>: $\mu_i = q_i - q_{i-1}$.

<u>Step 6</u>: calculate a_{ik}, $k = 0$, i from μ_i, σ_{i-1}^2, and $a_{i-1,k}$.

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

<u>Step 8</u>: $\sigma_i^2 = N_i/N_{i-1}$.

<u>Step 9</u>: Test σ_i^2. If $\sigma_i^2 \leq 0$, terminate the calculation with $n = i-1$ and set error flag.

<u>Step 10</u>: $i = i+1$, 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, LØCSIG

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

$$\text{IFMU} \begin{cases} \neq 0 \text{ print out all the quantities calculated by GETMUS} \\ = 0 \text{ do not print out data except in case of error } (\sigma_k^2 \leq 0). \end{cases}$$

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_i^2, 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,

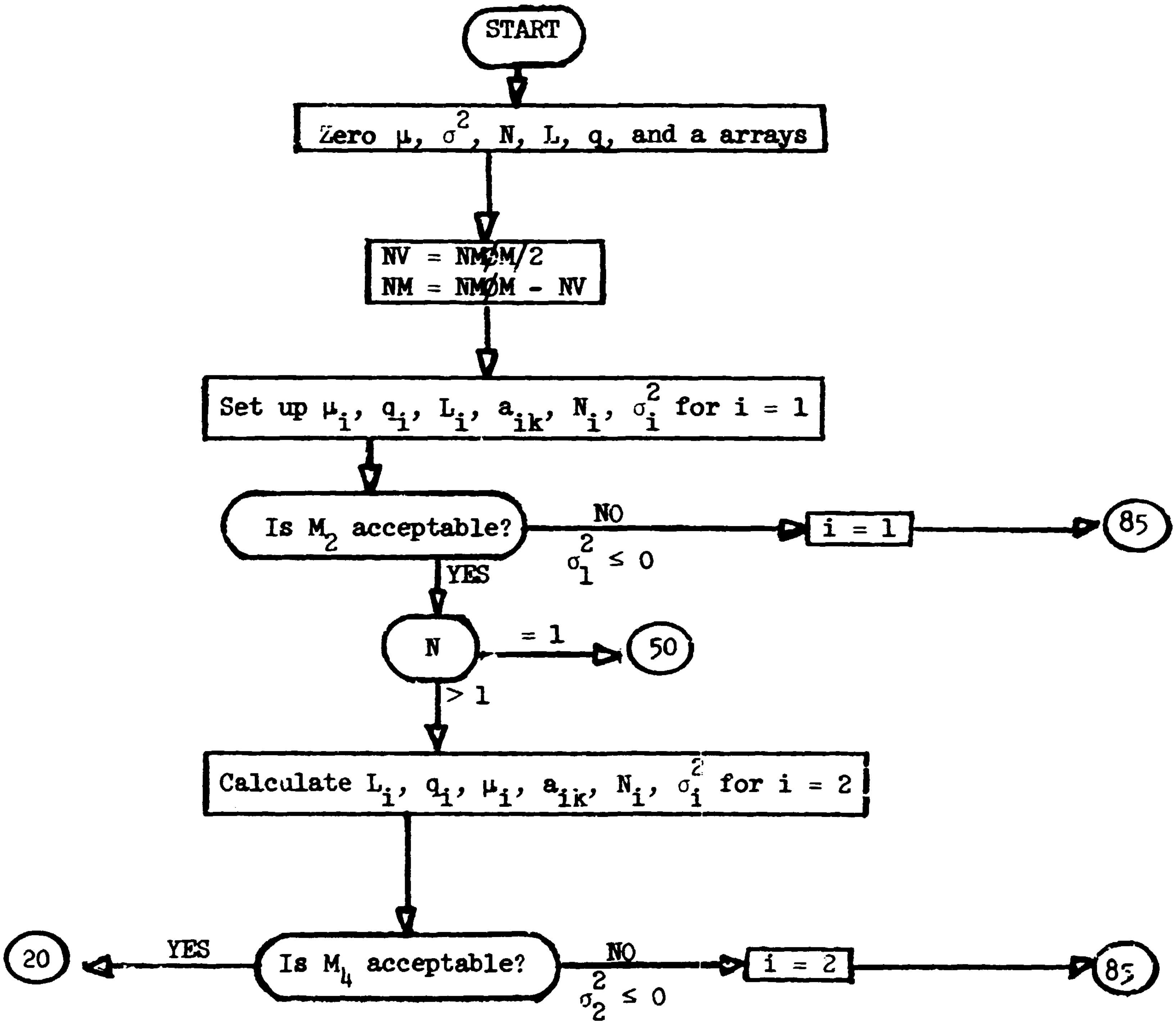
$Q(I) = q_i \quad i = 1, NM$

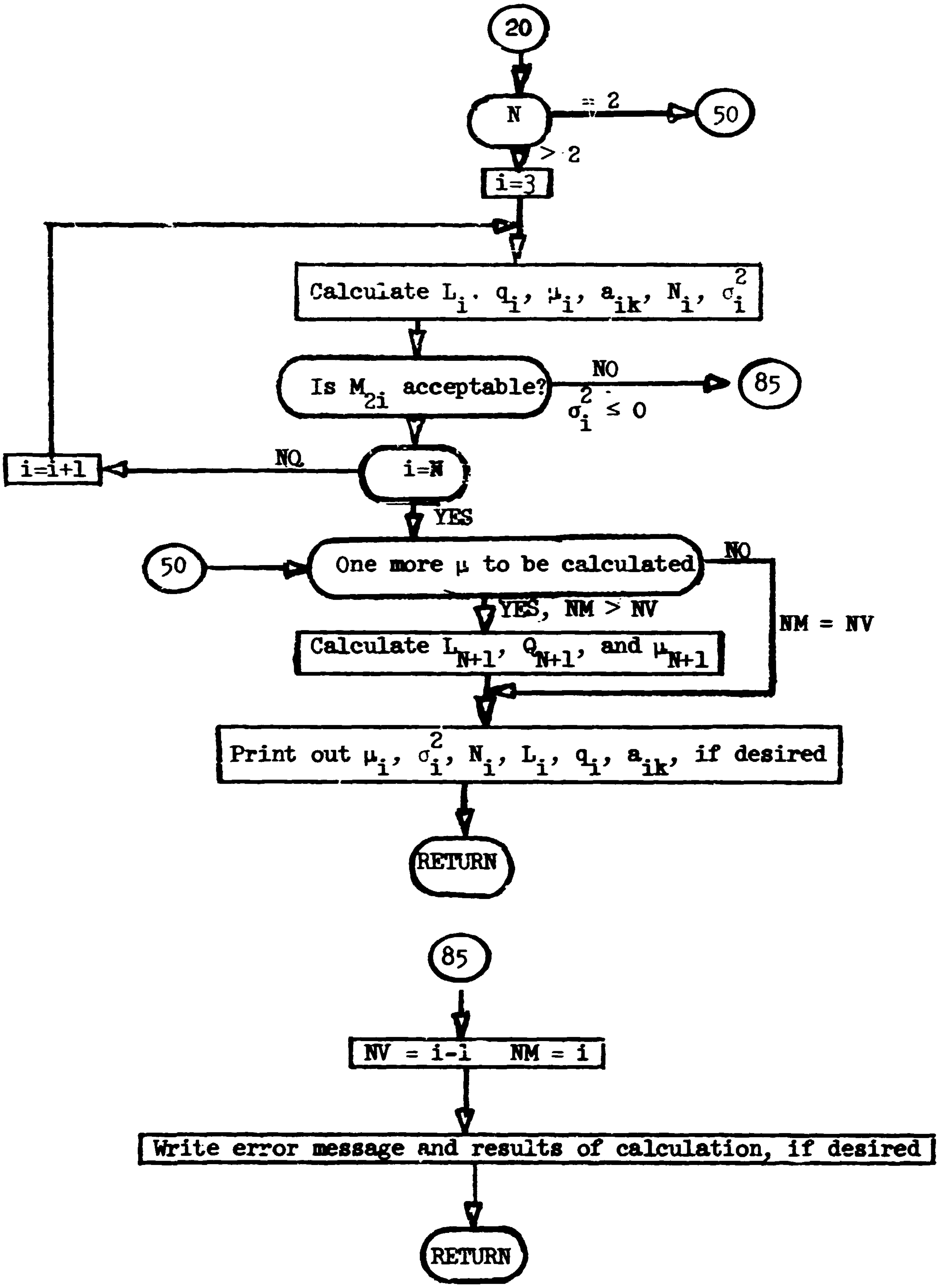
$A(I,K) = \mu_{i,K-1} \quad i = 1,NV; \ K = 1,i+1$

$L(I) = L_i \quad i = 1,NM.$

Limitations: NMØM $\leq$ 27.

Subroutine GETMUS


START
Zero μ, σ², N, L, q, and a arrays
NV = NMØM/2
NM = NMØM - NV
Set up μ_i, q_i, L_i, a_{ik}, N_i, σ_i^2 for i = 1
Is M_2 acceptable?
NO
$\sigma_1^2 \leq 0$
i = 1
85
YES
N
= 1
50
> 1
Calculate L_i, q_i, μ_i, a_{ik}, N_i, σ_i^2 for i = 2
Is M_4 acceptable?
YES
20
NO
$\sigma_2^2 \leq 0$
i = 2
85


20
N
= 2
50
> 2
i=3
Calculate L_i. q_i, μ_i, a_{ik}, N_i, σ_i^2
Is M_{2i} acceptable?
NO
$\sigma_i^2 \leq 0$
85
i=i+1
NO
i=N
YES
50
One more μ to be calculated
NO
YES, NM > NV
NM = NV
Calculate L_{N+1}, Q_{N+1}, and μ_{N+1}
Print out μ_i, σ_i^2, N_i, L_i, q_i, a_{ik}, if desired
RETURN
85
NV = i-1 NM = i
Write error message and results of calculation, if desired
RETURN

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

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:

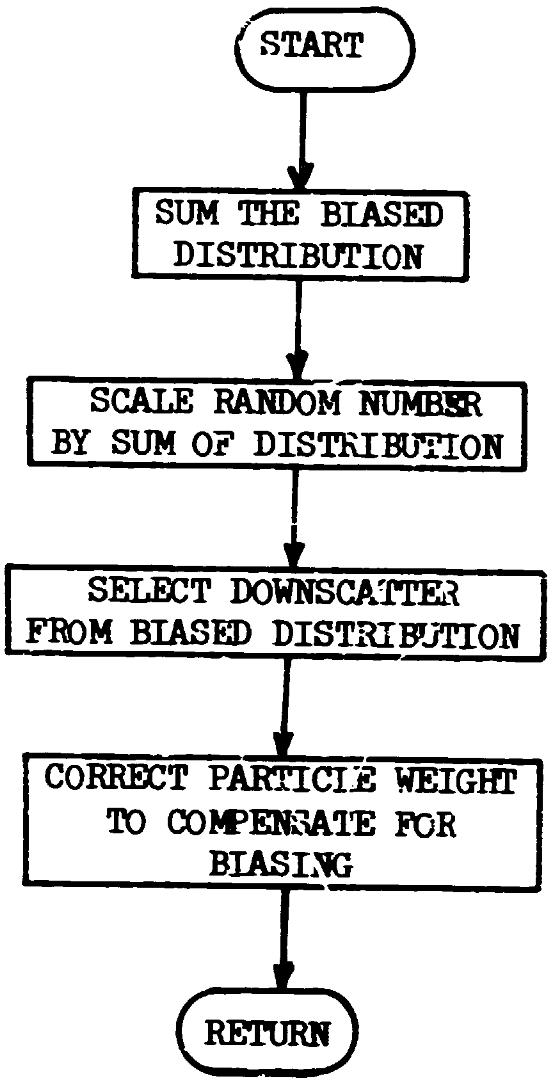
J - the number of downscattering groups,

WATE - modified to correct for the biasing.

Significant internal variables:

SBSIG is the normalization N of the biased distribution.

Subroutine GTIØUT


START
SUM THE BIASED DISTRIBUTION
SCALE RANDOM NUMBER BY SUM OF DISTRIBUTION
SELECT DOWNSCATTER FROM BIASED DISTRIBUTION
CORRECT PARTICLE WEIGHT TO COMPENSATE FOR BIASING
RETURN

Subroutine JNPUT

This subroutine is the executive routine for processing the cross sections from the ANISN or DTF-IV formats to the necessary probability tables. The major function of this routine is to mix the cross section 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 88

Variables changed: Blank common from ISTART to NMXSEC.

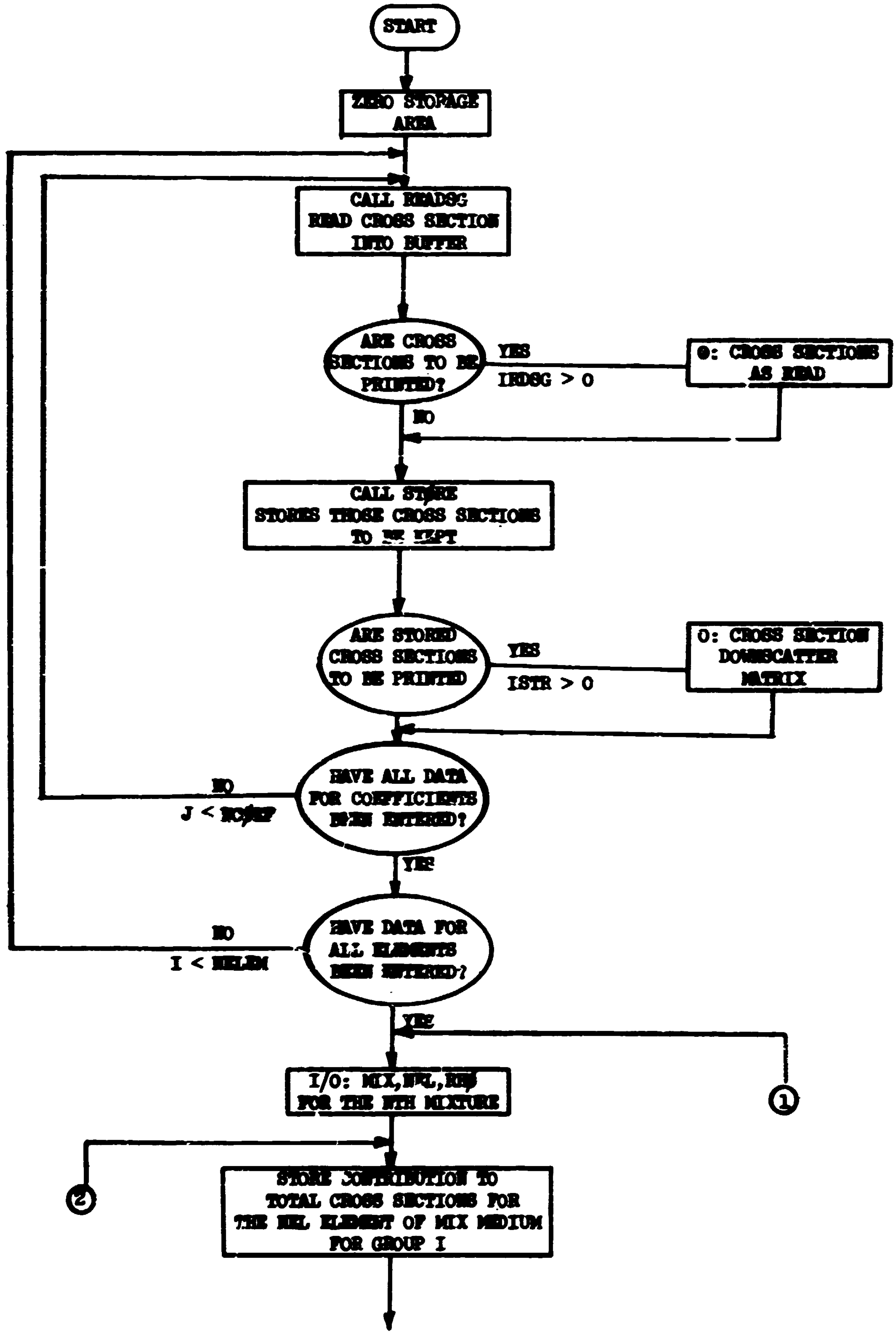
Input read: MIX NEL RHØ — RHØ times the cross section of the element NEL is added to the MIX medium cross section. If NEL is negative, the current mixing operation completes the cross section for that medium. There are NMIX of these cards read.

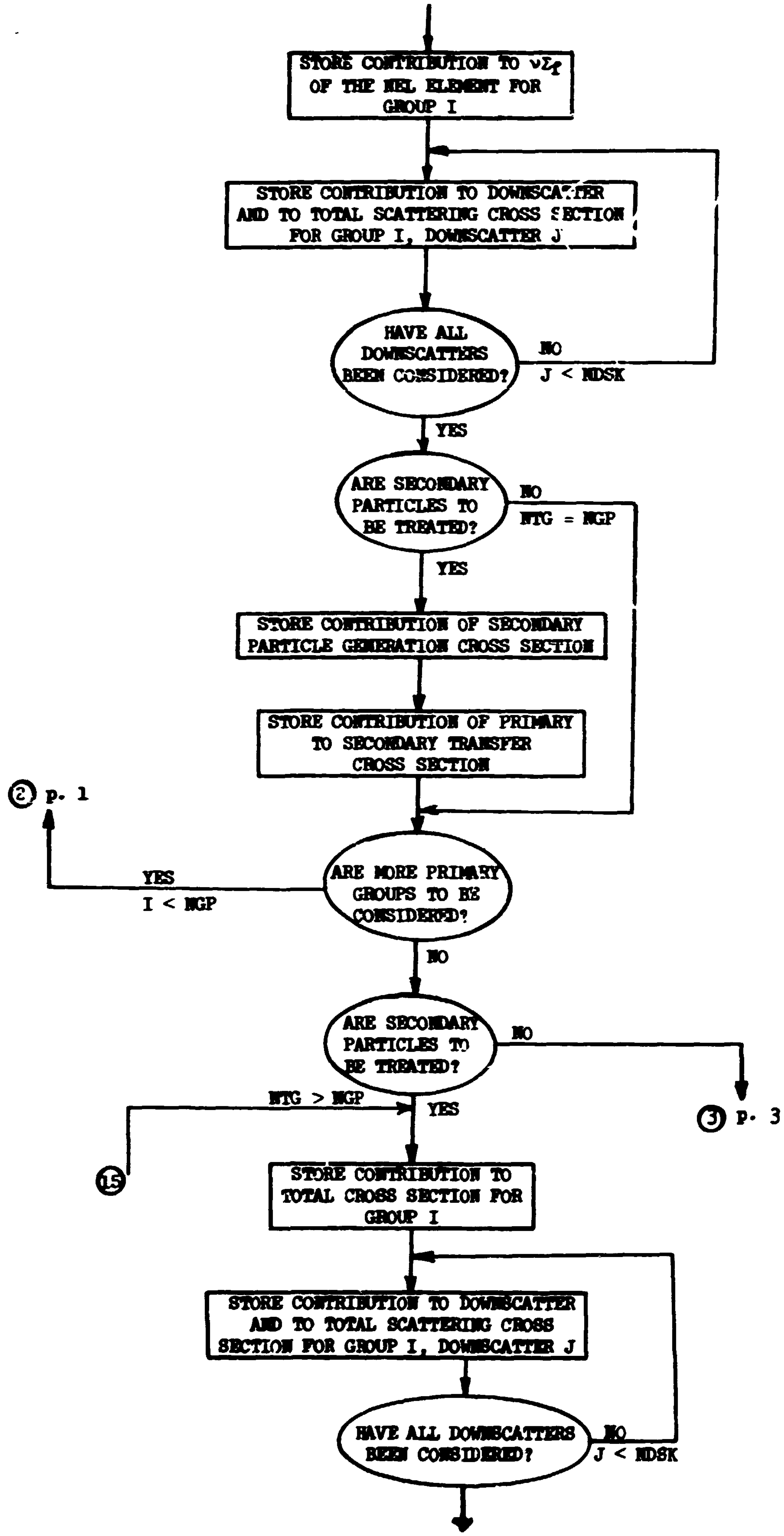
Significant internal variables:

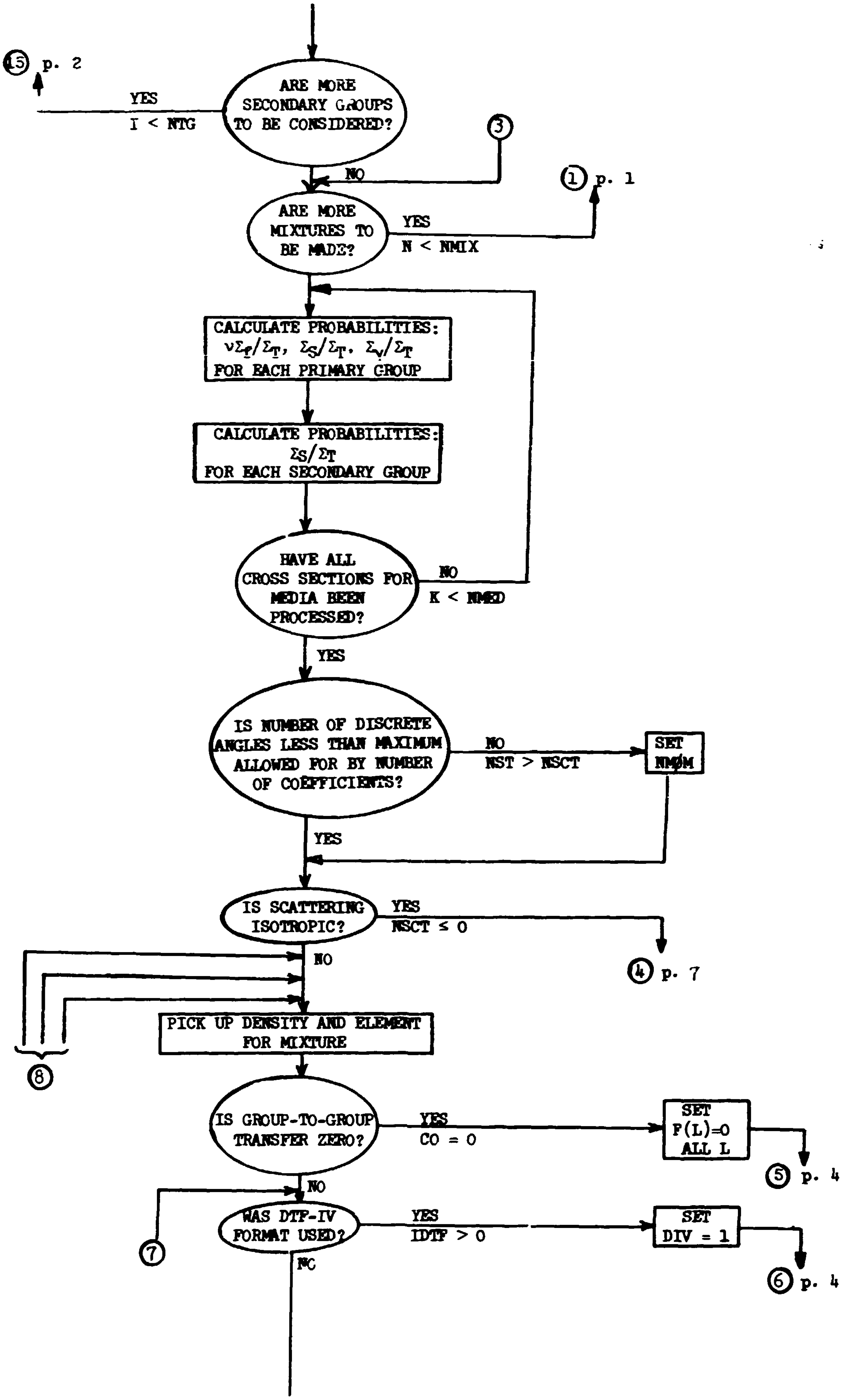
NDSK is the current number of downscatter groups for starting from present location.

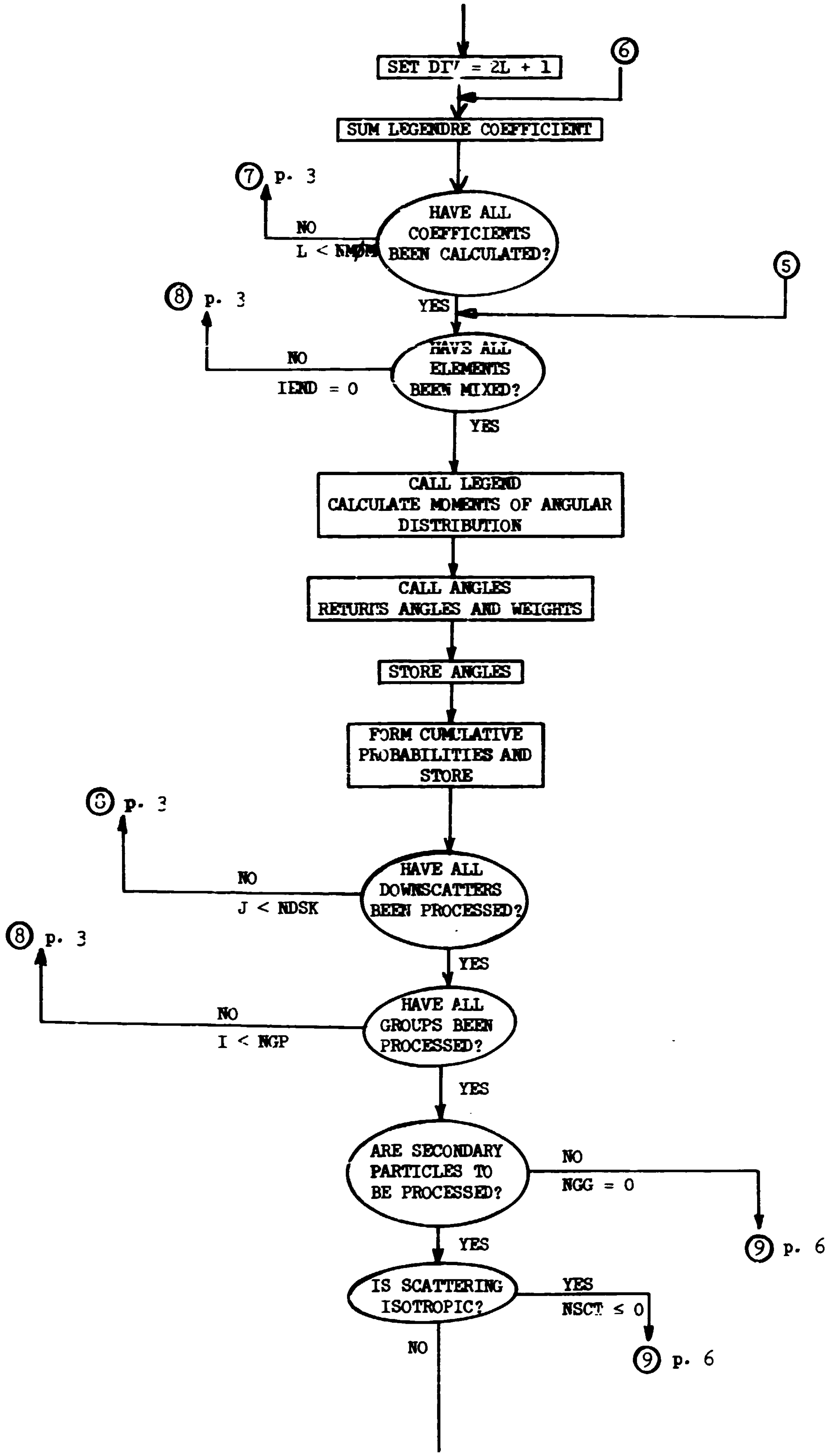
Limitations: If cross sections are to be mixed, and then the Legendre coefficients are to be restored, the first element must be mixed first. The first element should not appear in several media since the Legendre coefficients stored for medium 1 may write over the element 1 cross sections. The seriousness of this limitation is strongly dependent on the number of groups, coefficients, and elements.

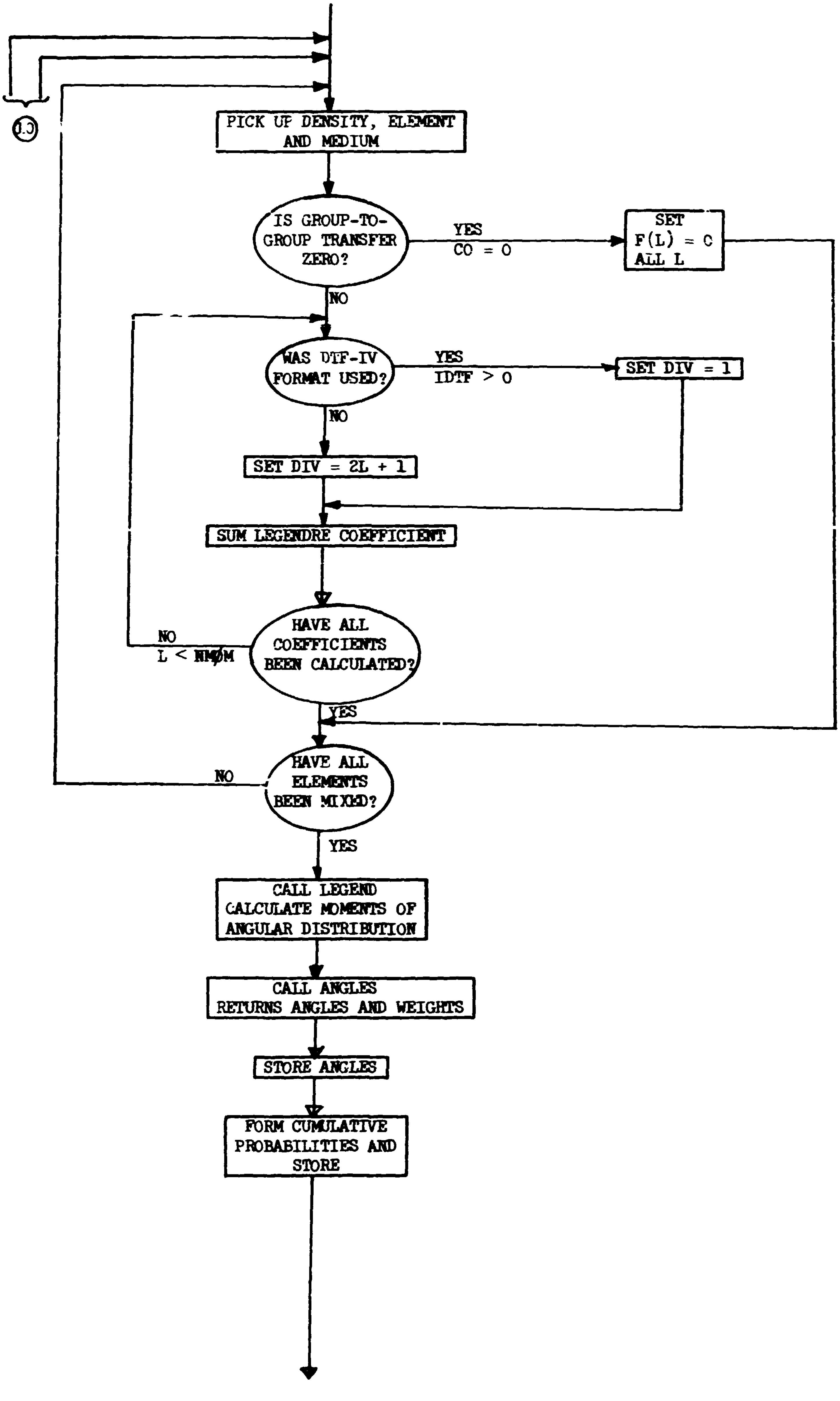
Subroutine JNPUT

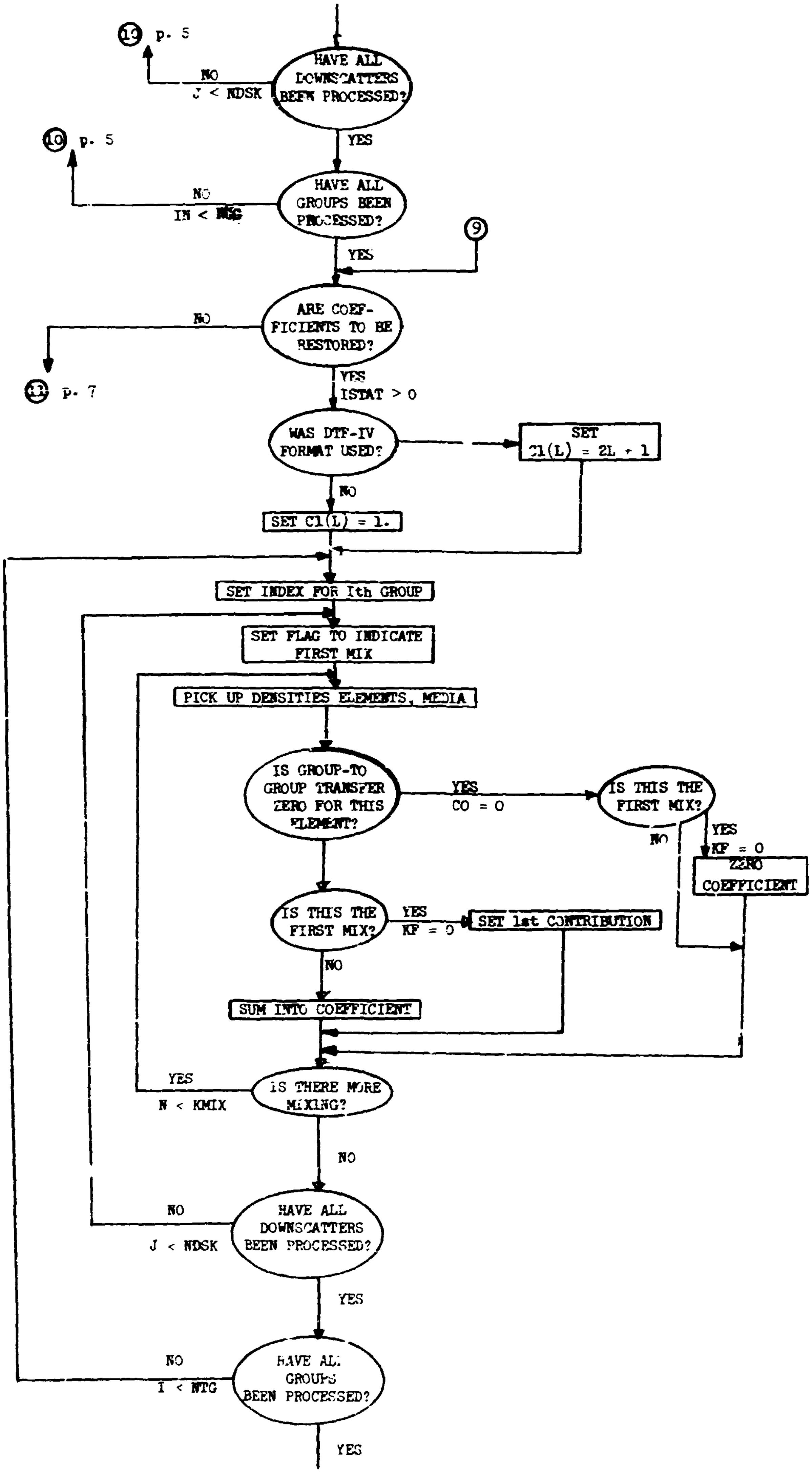

START
ZERO STORAGE AREA
CALL READSG
READ CROSS SECTION INTO BUFFER
ARE CROSS SECTIONS TO BE PRINTED?
YES
IRDSG > 0
O: CROSS SECTIONS AS READ
NO
CALL STØRE
STORES THOSE CROSS SECTIONS TO BE KEPT
ARE STORED CROSS SECTIONS TO BE PRINTED
YES
ISTR > 0
O: CROSS SECTION DOWNSCATTER MATRIX
HAVE ALL DATA FOR COEFFICIENTS BEEN ENTERED?
NO
J < NCØEF
YES
HAVE DATA FOR ALL ELEMENTS BEEN ENTERED?
NO
I < NELEM
YES
I/O: MIX,NEL,RHØ FOR THE NTH MIXTURE
1
2
STORE CONTRIBUTION TO TOTAL CROSS SECTIONS FOR THE NEL ELEMENT OF MIX MEDIUM FOR GROUP I


STORE CONTRIBUTION TO $\nu\Sigma_f$ OF THE NEL ELEMENT FOR GROUP I
STORE CONTRIBUTION TO DOWNSCATTER AND TO TOTAL SCATTERING CROSS SECTION FOR GROUP I, DOWNSCATTER J
HAVE ALL DOWNSCATTERS BEEN CONSIDERED?
NO
J < NDSK
YES
ARE SECONDARY PARTICLES TO BE TREATED?
NO
NTG = NGP
YES
STORE CONTRIBUTION OF SECONDARY PARTICLE GENERATION CROSS SECTION
STORE CONTRIBUTION OF PRIMARY TO SECONDARY TRANSFER CROSS SECTION
2 p. 1
YES
I < NGP
ARE MORE PRIMARY GROUPS TO BE CONSIDERED?
NO
ARE SECONDARY PARTICLES TO BE TREATED?
NO
3 p. 3
NTG > NGP
YES
15
STORE CONTRIBUTION TO TOTAL CROSS SECTION FOR GROUP I
STORE CONTRIBUTION TO DOWNSCATTER AND TO TOTAL SCATTERING CROSS SECTION FOR GROUP I, DOWNSCATTER J
HAVE ALL DOWNSCATTERS BEEN CONSIDERED?
NO
J < NDSK

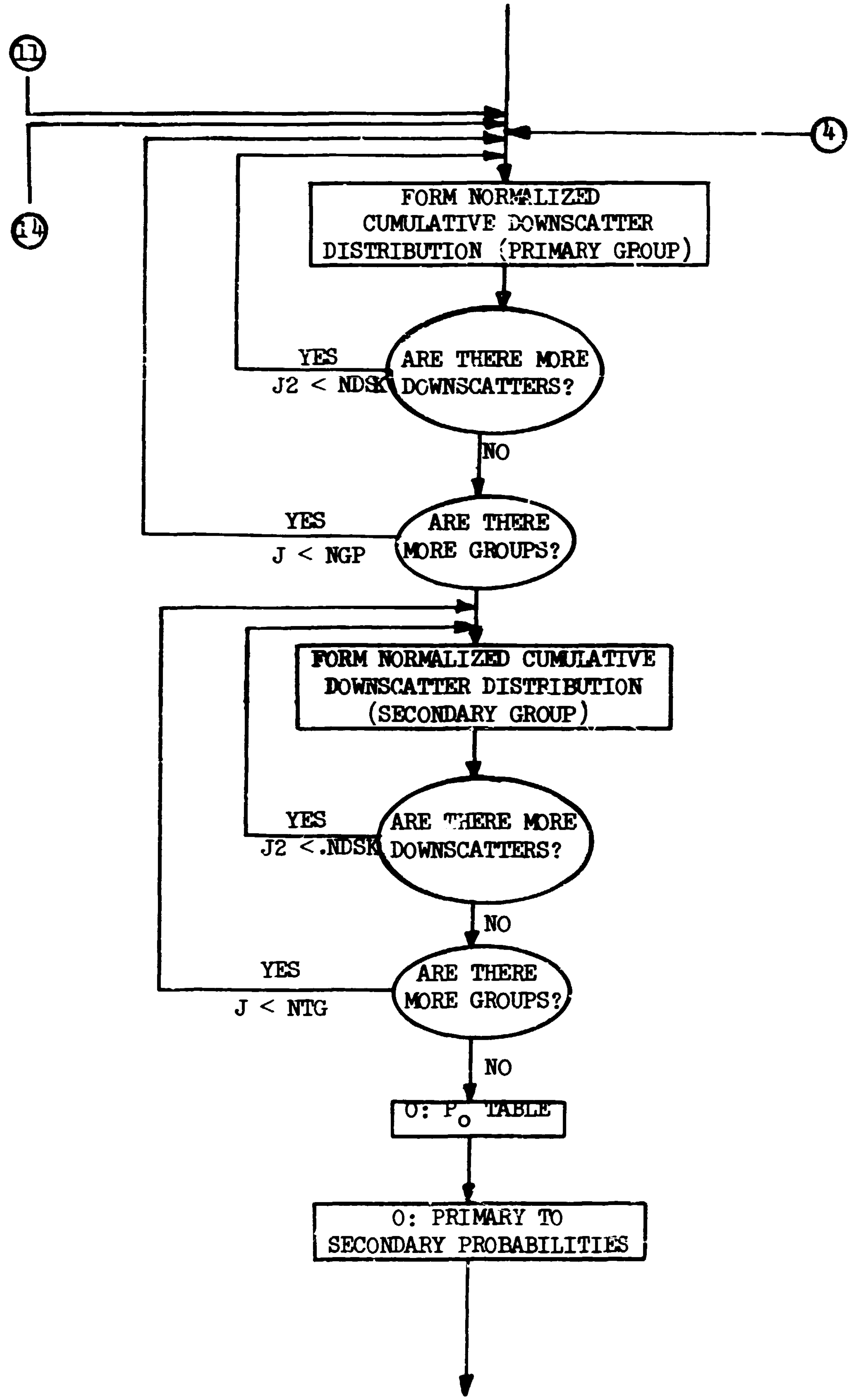

15 p. 2
YES
I < NTG
ARE MORE SECONDARY GROUPS TO BE CONSIDERED?
3
NO
1 p. 1
ARE MORE MIXTURES TO BE MADE?
YES
N < NMIX
CALCULATE PROBABILITIES: $\nu\Sigma_f/\Sigma_T$, Σ_S/Σ_T, Σ_v/Σ_T FOR EACH PRIMARY GROUP
CALCULATE PROBABILITIES: Σ_S/Σ_T FOR EACH SECONDARY GROUP
HAVE ALL CROSS SECTIONS FOR MEDIA BEEN PROCESSED?
NO
K < NMED
YES
IS NUMBER OF DISCRETE ANGLES LESS THAN MAXIMUM ALLOWED FOR BY NUMBER OF COEFFICIENTS?
NO
NST > NSCT
SET NMØM
YES
IS SCATTERING ISOTROPIC?
YES
NSCT ≤ 0
4 p. 7
NO
PICK UP DENSITY AND ELEMENT FOR MIXTURE
8
IS GROUP-TO-GROUP TRANSFER ZERO?
YES
CO = 0
SET F(L)=0 ALL L
5 p. 4
NO
7
WAS DTF-IV FORMAT USED?
YES
IDTF > 0
SET DIV = 1
6 p. 4
NO

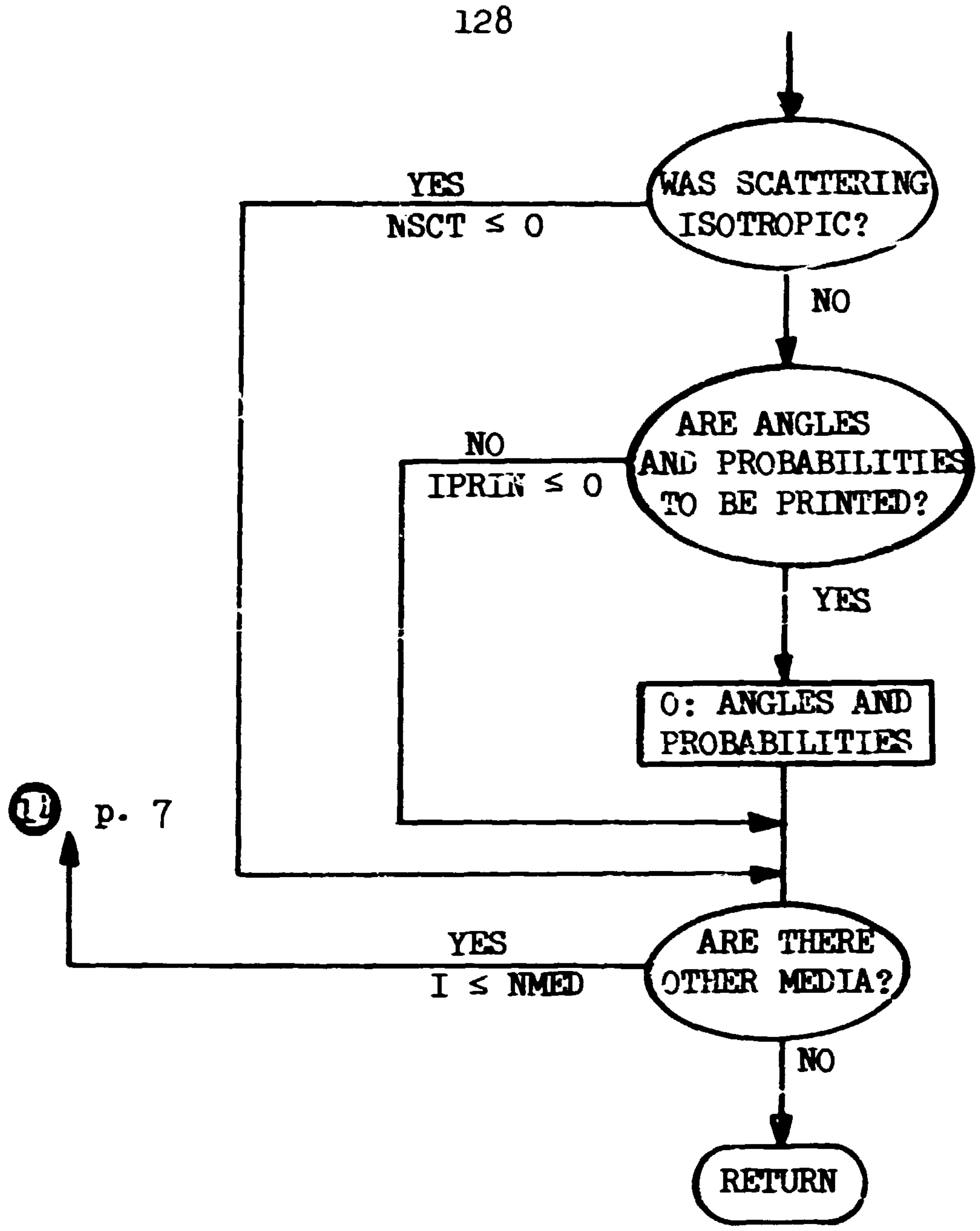

SET DI' = 2L + 1
6
SUM LEGENDRE COEFFICIENT
7 p. 3
NO
L < NMØM
HAVE ALL COEFFICIENTS BEEN CALCULATED?
5
YES
8 p. 3
NO
IEND = 0
HAVE ALL ELEMENTS BEEN MIXED?
YES
CALL LEGEND CALCULATE MOMENTS OF ANGULAR DISTRIBUTION
CALL ANGLES RETURNS ANGLES AND WEIGHTS
STORE ANGLES
FORM CUMULATIVE PROBABILITIES AND STORE
8 p. 3
NO
J < NDSK
HAVE ALL DOWNSCATTERS BEEN PROCESSED?
YES
8 p. 3
NO
I < NGP
HAVE ALL GROUPS BEEN PROCESSED?
YES
ARE SECONDARY PARTICLES TO BE PROCESSED?
NO
NGG = 0
9 p. 6
YES
IS SCATTERING ISOTROPIC?
YES
NSCT ≤ 0
9 p. 6
NO


10
PICK UP DENSITY, ELEMENT AND MEDIUM
IS GROUP-TO-GROUP TRANSFER ZERO?
YES
CO = 0
SET F(L) = C ALL L
NO
WAS DTF-IV FORMAT USED?
YES
IDTF > 0
SET DIV = 1
NO
SET DIV = 2L + 1
SUM LEGENDRE COEFFICIENT
HAVE ALL COEFFICIENTS BEEN CALCULATED?
NO
L < NMØM
YES
HAVE ALL ELEMENTS BEEN MIXED?
NO
YES
CALL LEGEND CALCULATE MOMENTS OF ANGULAR DISTRIBUTION
CALL ANGLES RETURNS ANGLES AND WEIGHTS
STORE ANGLES
FORM CUMULATIVE PROBABILITIES AND STORE



11 p. 5
NO
J < NDSK
HAVE ALL DOWNSCATTERS BEEN PROCESSED?
YES
10 p. 5
NO
IN < NGG
HAVE ALL GROUPS BEEN PROCESSED?
9
YES
NO
ARE COEF-FICIENTS TO BE RESTORED?
11 p. 7
YES
ISTAT > 0
WAS DTF-IV FORMAT USED?
SET C1(L) = 2L + 1
NO
SET C1(L) = 1.
SET INDEX FOR Ith GROUP
SET FLAG TO INDICATE FIRST MIX
PICK UP DENSITIES ELEMENTS, MEDIA
IS GROUP-TO GROUP TRANSFER ZERO FOR THIS ELEMENT?
YES
CO = 0
IS THIS THE FIRST MIX?
NO
YES
KF = 0
ZERO COEFFICIENT
IS THIS THE FIRST MIX?
YES
KF = 0
SET 1st CONTRIBUTION
NO
SUM INTO COEFFICIENT
YES
N < KMIX
IS THERE MORE MIXING?
NO
NO
J < NDSK
HAVE ALL DOWNSCATTERS BEEN PROCESSED?
YES
NO
I < NTG
HAVE ALL GROUPS BEEN PROCESSED?
YES


11
4
14
FORM NORMALIZED CUMULATIVE DOWNSCATTER DISTRIBUTION (PRIMARY GROUP)
YES
J2 < NDSK
ARE THERE MORE DOWNSCATTERS?
NO
YES
J < NGP
ARE THERE MORE GROUPS?
FORM NORMALIZED CUMULATIVE DOWNSCATTER DISTRIBUTION (SECONDARY GROUP)
YES
J2 < NDSK
ARE THERE MORE DOWNSCATTERS?
NO
YES
J < NTG
ARE THERE MORE GROUPS?
NO
0: P_O TABLE
0: PRIMARY TO SECONDARY PROBABILITIES


WAS SCATTERING ISOTROPIC?
YES
NSCT ≤ 0
NO
ARE ANGLES AND PROBABILITIES TO BE PRINTED?
NO
IPRIN ≤ 0
YES
0: ANGLES AND PROBABILITIES
11
p. 7
YES
I ≤ NMED
ARE THERE OTHER MEDIA?
NO
RETURN

Subroutine LEGEND

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

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

The output of LEGEND consists of the moments,

$$M_n = \int_{-1}^{1} \mu^n f(\mu)d\mu \quad n = 1,NM\emptyset M .$$

Method: If we let

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

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

$$\begin{aligned} P^{-1}_{n,\ell} &= \frac{1}{2} \int_{-1}^{1} \mu^{n-1}[(2\ell + 1)\mu\, P_\ell(\mu)]d\mu \\ &= \frac{1}{2} \int_{-1}^{1} \mu^{n-1}[(\ell + 1)P_{\ell+1}(\mu) + \ell\, P_{\ell-1}(\mu)]\, d\mu \\ &= \frac{\ell + 1}{2} \int_{-1}^{1} \mu^{n-1}\, P_{\ell+1}(\mu)d\mu + \frac{\ell}{2} \int_{-1}^{1} \mu^{n-1}\, P_{\ell-1}(\mu)d\mu \\ &= \frac{\ell + 1}{2\ell + 3} P^{-1}_{n-1,\ell+1} + \frac{\ell}{2\ell - 1} P^{-1}_{n-1,\ell-1} . \end{aligned}$$

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

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

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

$$= \sum_{\ell=0}^{n} P^{-1}_{n\ell} f_\ell \ .$$

Called from: JNPUT

Commons required: MØMENT

Variables required:

NMØM

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

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

Significant internal variables:

$P1(\ell) = P^{-1}_{n-1,\ell}$

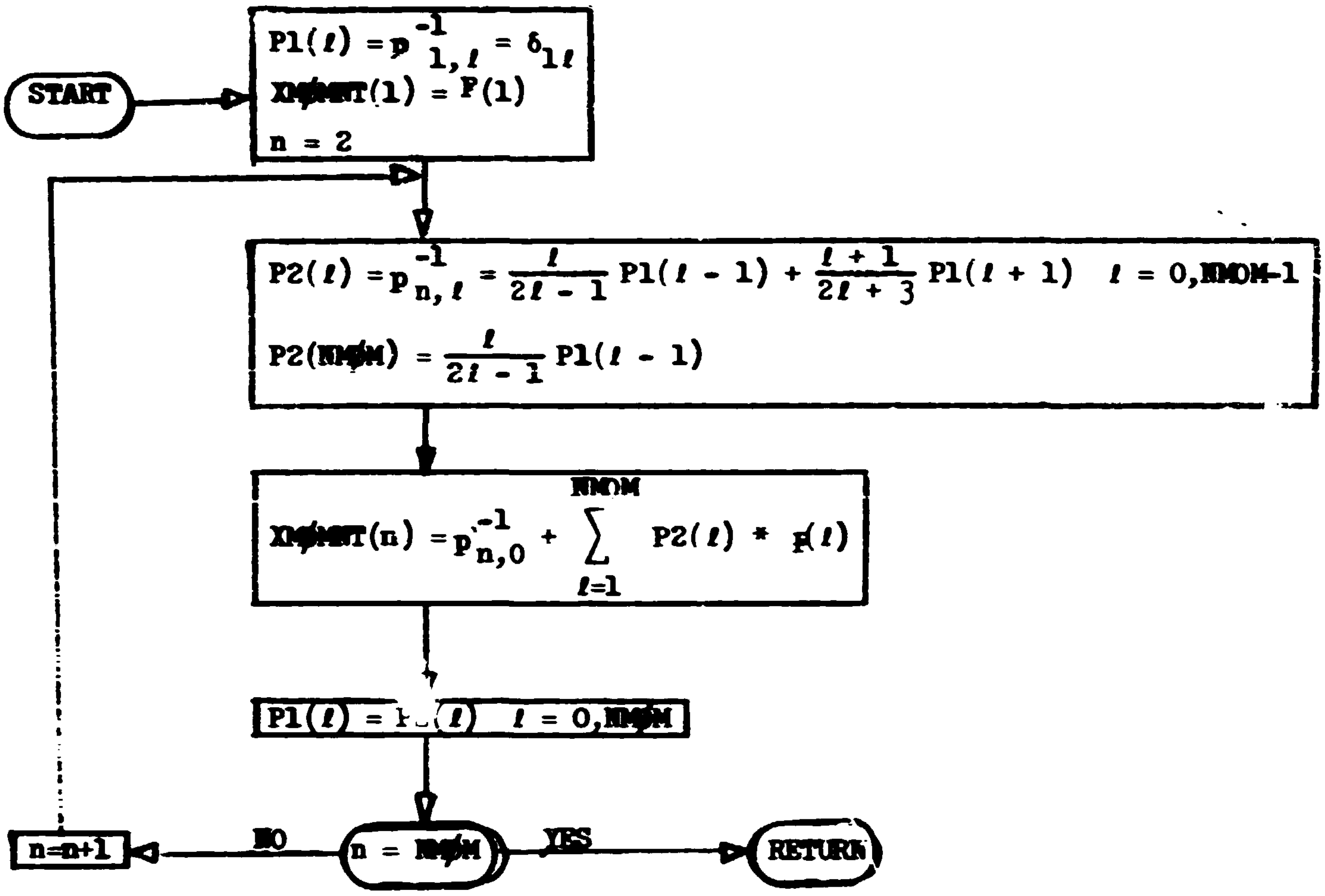
$P2(\ell) = P^{-1}_{n,\ell}$

$P10 = P^{-1}_{n-1,0}$

$P20 = P^{-1}_{n,\ell}$.

Limitations: NMØM $\leq$ 24.

Subroutine LEGEND


START
$P1(\ell) = p^{-1}_{1,\ell} = \delta_{1\ell}$
XMØMNT(1) = F(1)
n = 2
$P2(\ell) = p^{-1}_{n,\ell} = \frac{\ell}{2\ell - 1} P1(\ell - 1) + \frac{\ell + 1}{2\ell + 3} P1(\ell + 1) \quad \ell = 0, NMOM-1$
$P2(NMØM) = \frac{\ell}{2\ell - 1} P1(\ell - 1)$
$XMØMNT(n) = p^{-1}_{n,0} + \sum_{\ell=1}^{NMOM} P2(\ell) * F(\ell)$
$P1(\ell) = P2(\ell) \quad \ell = 0, NMØM$
n = NMØM
NO
YES
n=n+1
RETURN

Note: Recursion relation terms involving zeroth order coefficients must be handled separately. P10 is P1(0), etc.

Subroutine MAMENT (NMØ)

This routine converts moments to Legendre coefficients. The moments

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

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

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

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

As this is an identity, we may separately equate the coefficients 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:

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

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

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

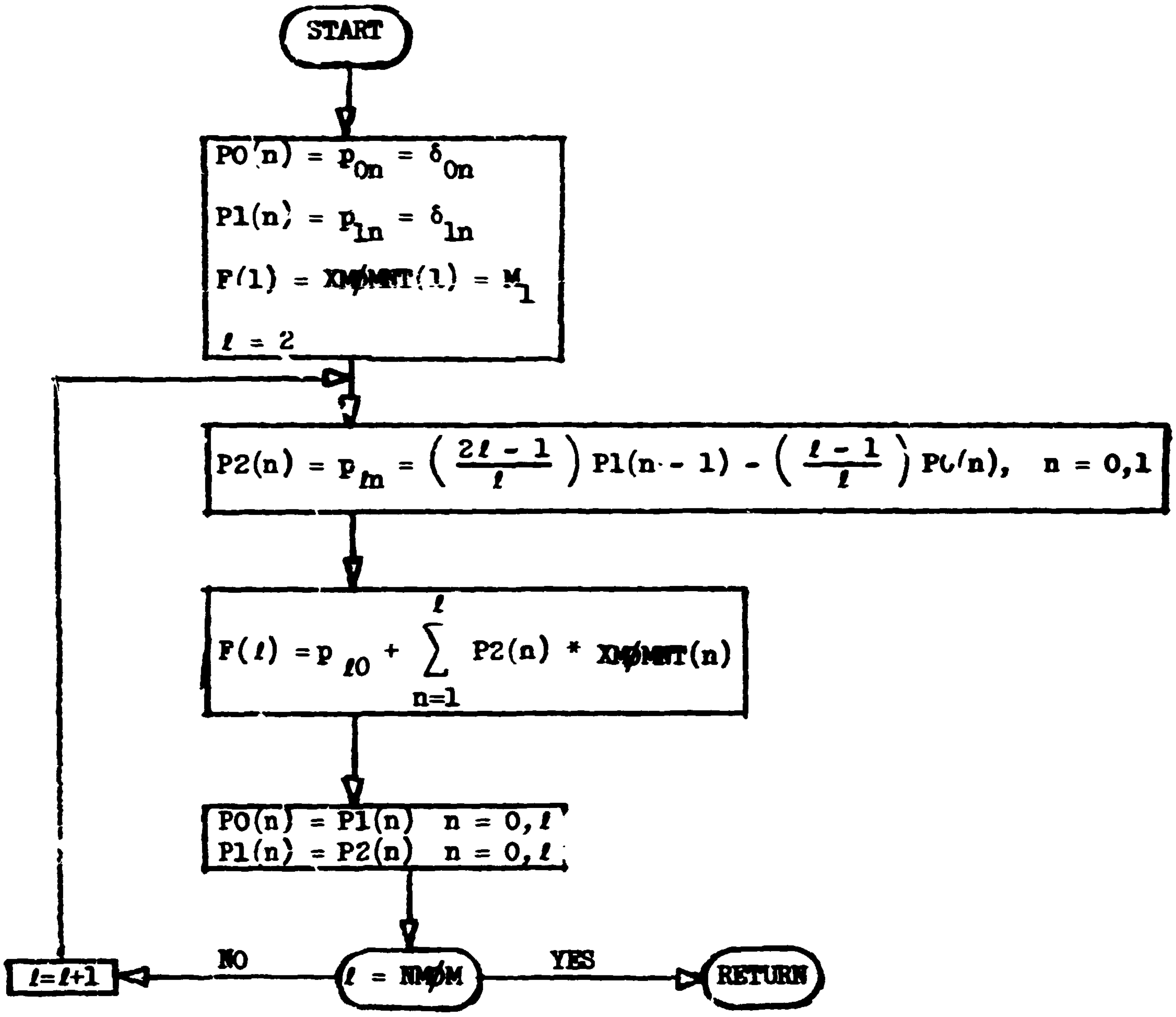
$P00 = P_{\ell-2,0}$

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

$P20 = P_{\ell 0}$

Limitations: NMØ $\leq$ 25.

Subroutine MAMENT


START
$P0(n) = p_{0n} = \delta_{0n}$
$P1(n) = p_{1n} = \delta_{1n}$
$F(1) = XMØMNT(1) = M_1$
$\ell = 2$
$P2(n) = p_{\ell n} = \left(\frac{2\ell - 1}{\ell}\right) P1(n - 1) - \left(\frac{\ell - 1}{\ell}\right) P0(n), \quad n = 0,1$
$F(\ell) = p_{\ell 0} + \sum_{n=1}^{\ell} P2(n) * XMØMNT(n)$
$P0(n) = P1(n) \quad n = 0,\ell$
$P1(n) = P2(n) \quad n = 0,\ell$
$\ell = \ell+1$
NO
ℓ = NMØM
YES
RETURN

Note: Recursion relations involving zeroth order coefficients must be handled separately. P00 is P0(0), etc.

Subroutine NSIGTA (IGA, JMED, TSIG, PNAB)

The function of this subroutine is to look up the total cross section and non-absorption probability for energy group IGA and geometry medium JMED.

Called from: EUCLID, NXTCØL, User routines

Subroutines called: GTMED

Commons required: Blank, LØCSIG

Variables required:

ISPØRG, ISTART, INABØG - from common LØCSIG, see page 88,

IGA - energy group,

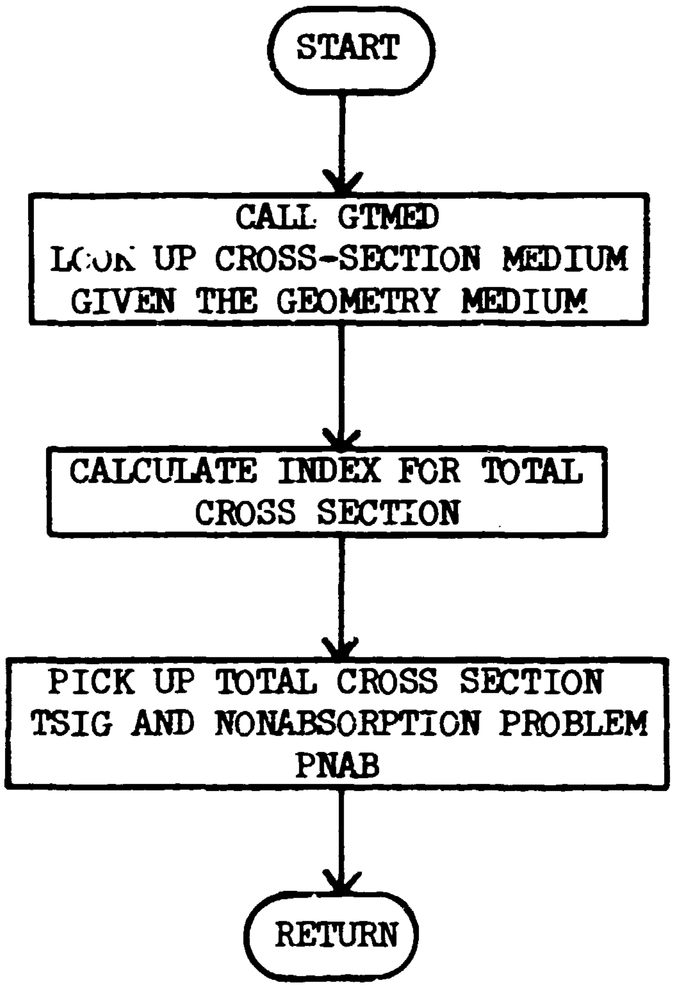
JMED - geometry medium.

Variables changed:

TSIG - total cross section,

PNAB - non-absorption probability.

Subroutine NSIGTA


START
CALL GTMED
LOOK UP CROSS-SECTION MEDIUM
GIVEN THE GEOMETRY MEDIUM
CALCULATE INDEX FOR TOTAL
CROSS SECTION
PICK UP TOTAL CROSS SECTION
TSIG AND NONABSORPTION PROBLEM
PNAB
RETURN

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 of

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

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 88)

Variables changed:

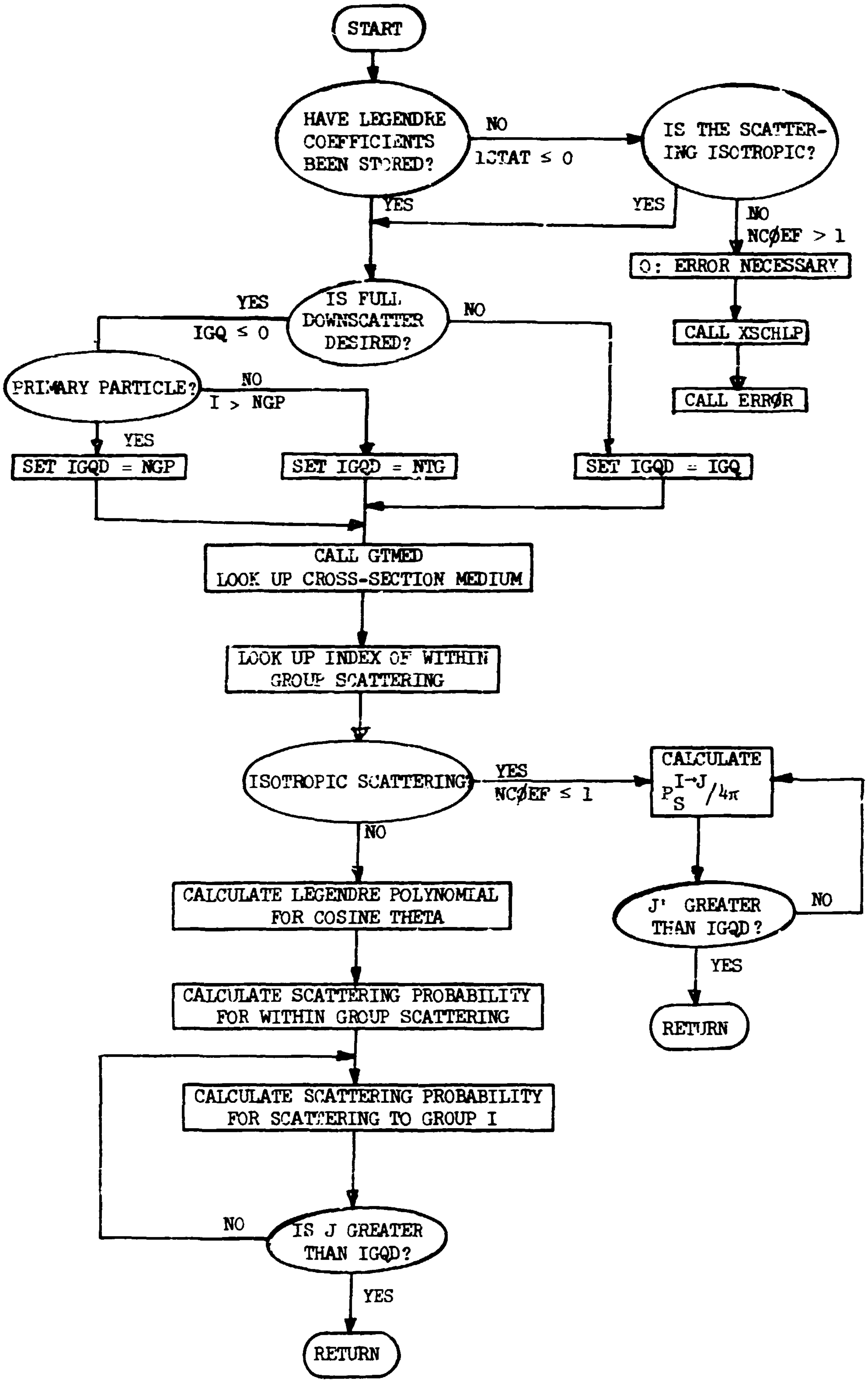
PMU - the probability of scattering through an angle whose cosine is θ; PMU is dimensioned by NMTG,

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.


START
HAVE LEGENDRE COEFFICIENTS BEEN STORED?
NO
ISTAT ≤ 0
IS THE SCATTERING ISOTROPIC?
YES
YES
NO
NCØEF > 1
Q: ERROR NECESSARY
CALL XSCHLP
CALL ERRØR
IS FULL DOWNSCATTER DESIRED?
YES
IGQ ≤ 0
NO
PRIMARY PARTICLE?
NO
I > NGP
YES
SET IGQD = NGP
SET IGQD = NTG
SET IGQD = IGQ
CALL GTMED
LOOK UP CROSS-SECTION MEDIUM
LOOK UP INDEX OF WITHIN GROUP SCATTERING
ISOTROPIC SCATTERING?
YES
NCØEF ≤ 1
CALCULATE $P_S^{I\to J}/4\pi$
J' GREATER THAN IGQD?
NO
YES
RETURN
NO
CALCULATE LEGENDRE POLYNOMIAL FOR COSINE THETA
CALCULATE SCATTERING PROBABILITY FOR WITHIN GROUP SCATTERING
CALCULATE SCATTERING PROBABILITY FOR SCATTERING TO GROUP I
NO
IS J GREATER THAN IGQD?
YES
RETURN

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 - value of the argument desired,

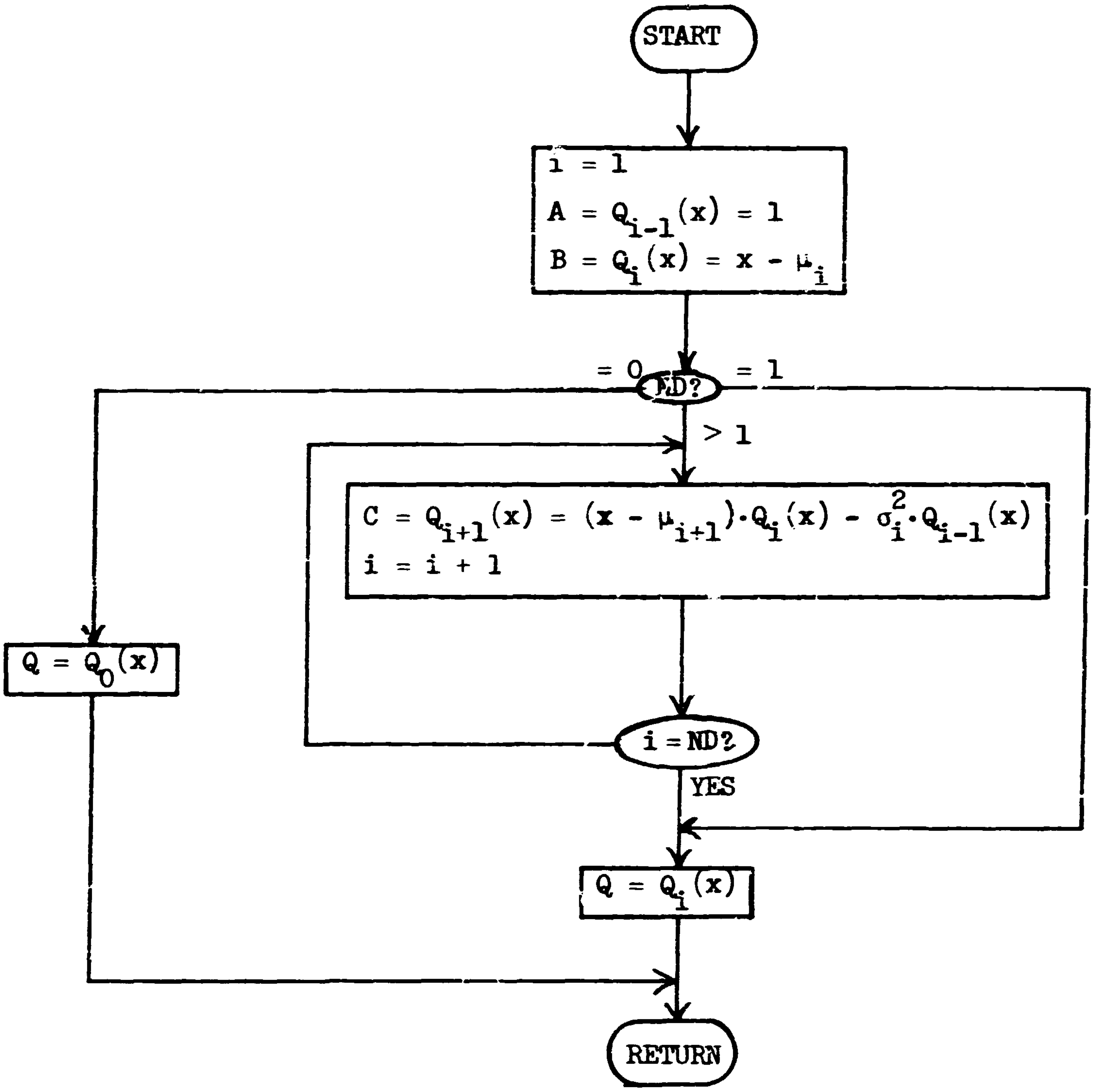
$\left.\begin{array}{l} \text{XMU(i)} = \mu_i \\ \text{VAR(i)} = \sigma^2_i \end{array}\right\}$ in labelled common MEANS

Variables changed:

Q - the value of the function.

Limitations: ND $\leq$ 14.

Function Q (ND,X)



Subroutine READSG

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 format cross sections may be read; otherwise, the ANISN format is assumed.

The ANISN cross-section format makes use of the repeat feature; thus, there is a mixture of Hollerith and numbers on the card. This subroutine will therefore be different for various computers. On IBM machines each cross-section card is read twice; once for the Hollerith R and once for the cross-section values. For CDC machines decode may be used to separate the Hollerith. DTF-IV format does not permit repeats, 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. Three possibilities are taken into account: (1) an energy group number in columns 73-76 and a sequence number for that group in columns 77-80 (format from codes such as GAM or MUG); (2) same as (1) except columns 73-76 are blank (format from codes such as SUPERTOG or XSDRN); (3) columns 73-76 are blank and columns 77-80 contain a card sequence number starting at 1 for each set of cross sections (format from ANISN). Non-numeric characters, including blanks, may precede or follow the above sequence numbers without affecting the checks.

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 IXTAPE > 0 then 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 desired cross sections must be required in the order in which they are on tape, and the element identifiers must be the fourth integer in the identification record. These identification numbers are required on input card D of the cross-section input.

Called from: JNPUT

Subroutines called:

XSCHLP
FETYPE
BCDTØI

} Library functions at Oak Ridge National Laboratory to determine if a Hollerith is a number or a letter and to convert EBCDIC to integer, respectively.

Commons required: Blank, LØCSIG

Variables required: INPBUF, INGP, INDS, KKK, IXTAPE, IDTF (from common LØCSIG, see page 88)

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

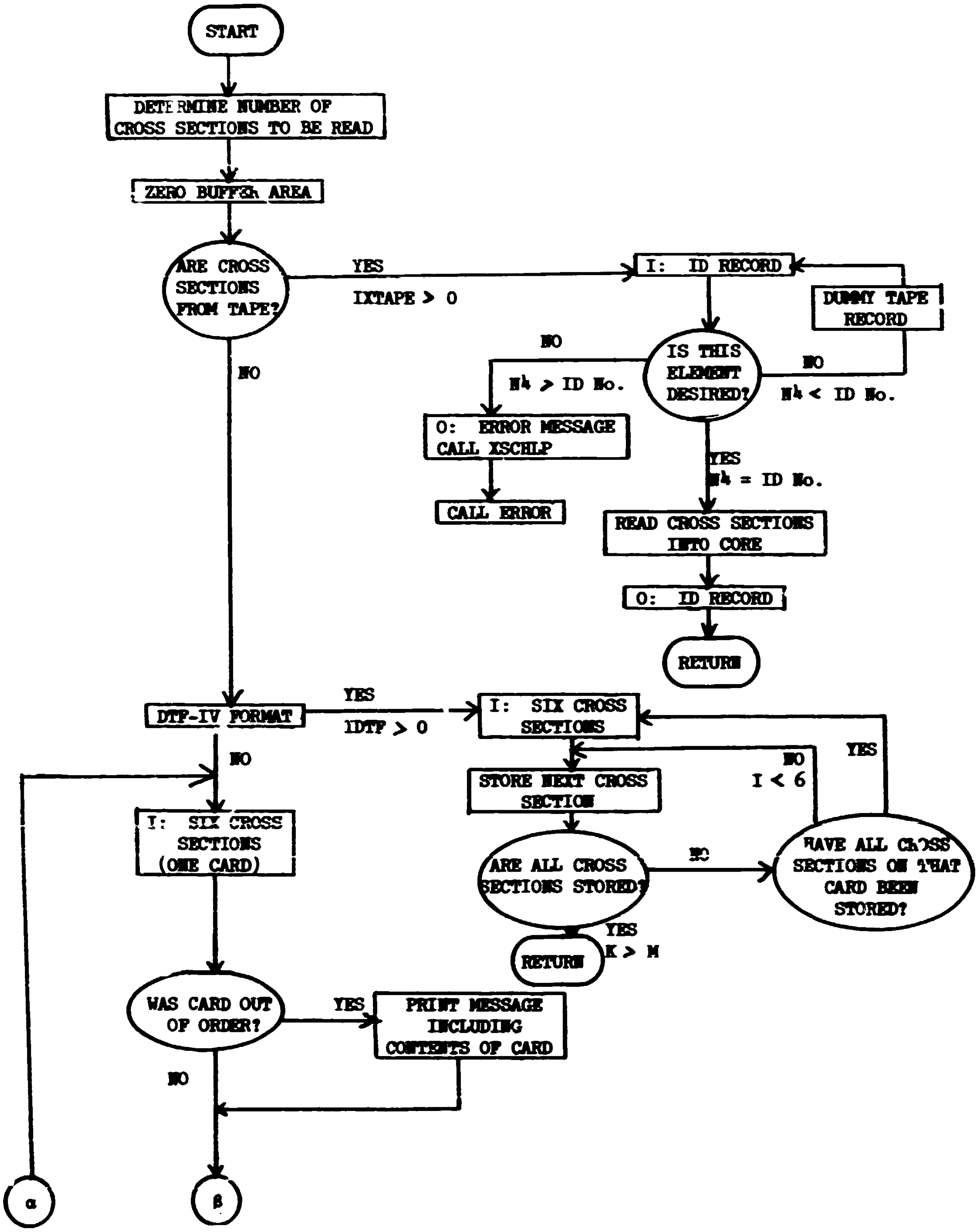
Significant internal variables:

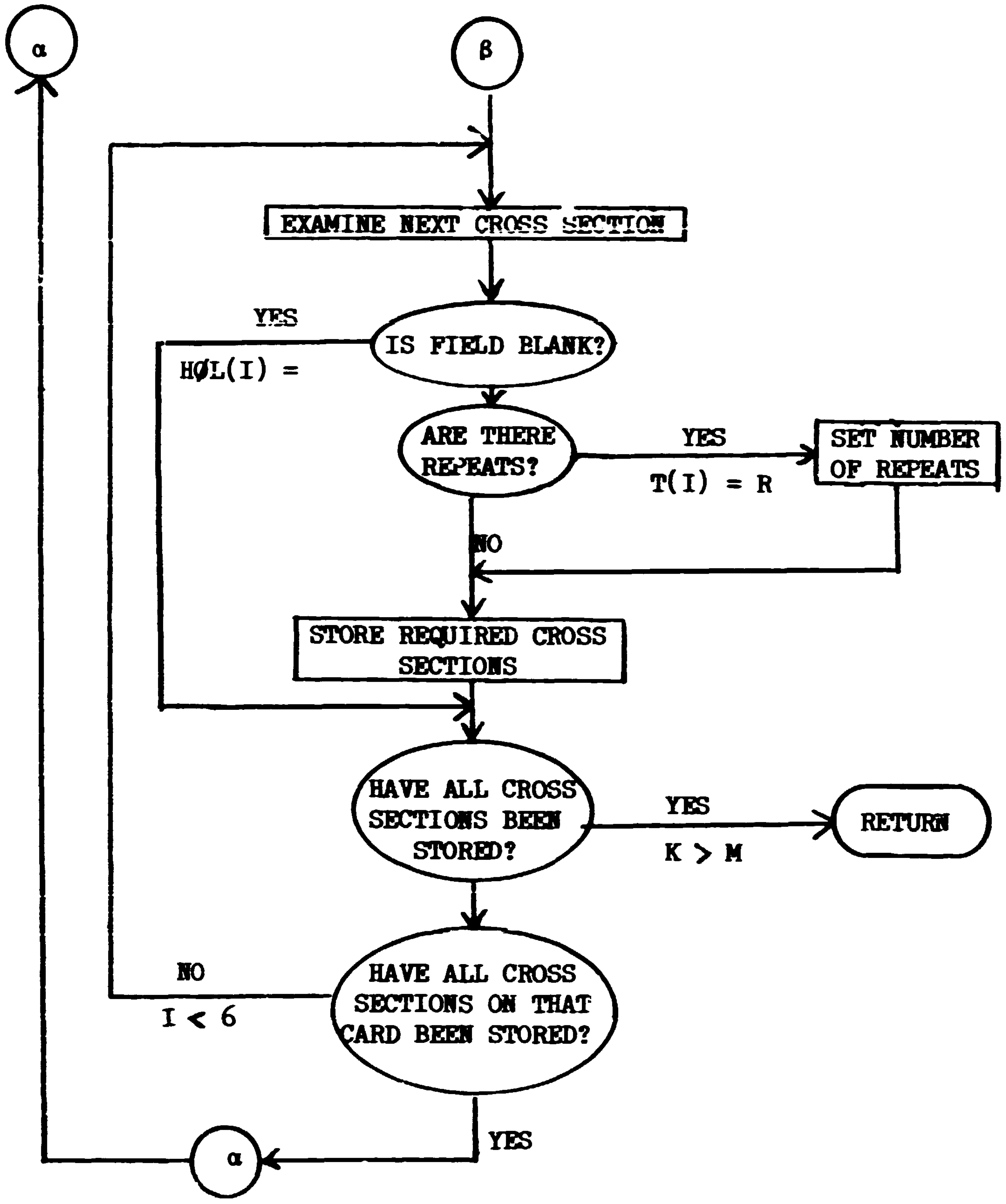
M - number of cross sections for each coefficient,

NP - number of repeats for a particular cross section.

Limitations: Card formats must be either ANISN or DTF-IV, or a binary tape may be used.

Subroutine READSG




α
β
EXAMINE NEXT CROSS SECTION
IS FIELD BLANK?
YES
HØL(I) =
ARE THERE REPEATS?
YES
T(I) = R
SET NUMBER OF REPEATS
NO
STORE REQUIRED CROSS SECTIONS
HAVE ALL CROSS SECTIONS BEEN STORED?
YES
K > M
RETURN
NO
I < 6
HAVE ALL CROSS SECTIONS ON THAT CARD BEEN STORED?
YES
α

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: INPUT

Commons required: Blank, LØCSIG

Variables required: IE - element number

IC - coefficient number

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

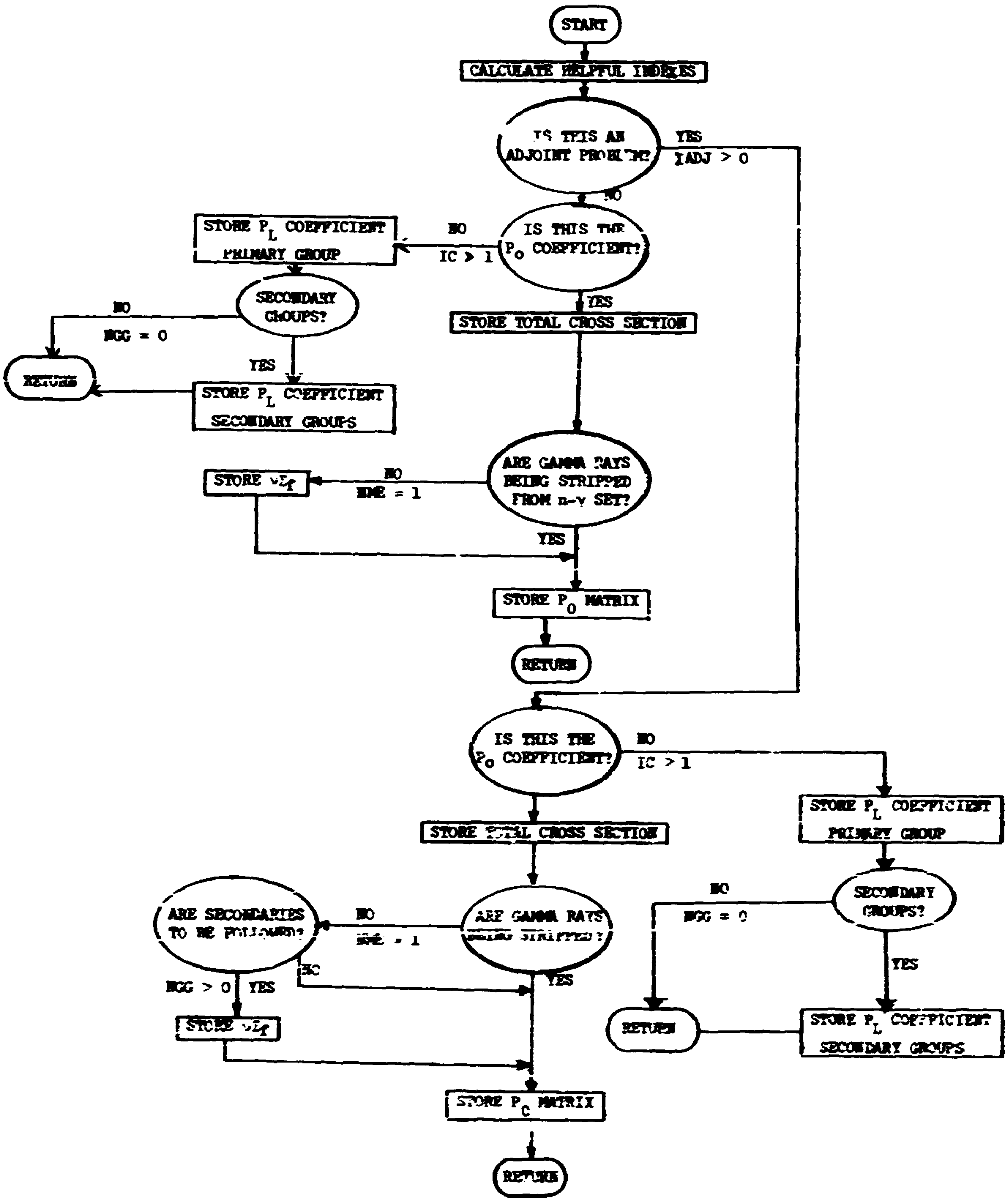
INPBUF, NTG, NTS NCØEF, ISPØRT, INFPØG, ISIGØG, INDS, IADJ, NME (from common LØCSIG, see page 88).

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.


START
CALCULATE HELPFUL INDEXES
IS THIS AN ADJOINT PROBLEM?
YES
IADJ > 0
NO
IS THIS THE P_o COEFFICIENT?
NO
IC > 1
STORE P_L COEFFICIENT PRIMARY GROUP
SECONDARY GROUPS?
NO
NGG = 0
RETURN
YES
STORE P_L COEFFICIENT SECONDARY GROUPS
YES
STORE TOTAL CROSS SECTION
ARE GAMMA RAYS BEING STRIPPED FROM n-γ SET?
NO
NME = 1
STORE $\nu\Sigma_f$
YES
STORE P_0 MATRIX
RETURN
IS THIS THE P_o COEFFICIENT?
NO
IC > 1
STORE P_L COEFFICIENT PRIMARY GROUP
SECONDARY GROUPS?
NO
NGG = 0
YES
RETURN
STORE P_L COEFFICIENT SECONDARY GROUPS
STORE TOTAL CROSS SECTION
ARE GAMMA RAYS BEING STRIPPED?
NO
NME = 1
ARE SECONDARIES TO BE FOLLOWED?
NO
NGG > 0
YES
YES
STORE $\nu\Sigma_f$
STORE P_C MATRIX
RETURN

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.

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

Called from: INPUT

Subroutines called: JNPUT, ALBIN, XSCHLP

Commons required: Blank, LØCSIG

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 cell used in blank common before XSEC was called.

Input: There are four cards read by subroutine XSEC. These cards contain:*

First Card - comment card,

Second Card - NGP, NDC, NGG, NDSG, INGP, INDS, NMED, NELEM, NMIX, NCØETF, NSCT, ISTAT, IXTAPE. For definitions see common LØCSIG, page 88.

*A more detailed description is given in Appendix C.

Third Card - IRDSG, ISTR, IFMU, IMØM, IPRIN, IPUN, IDTF. For definitions see common LØCSIG, page 88.

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

Variables changed:

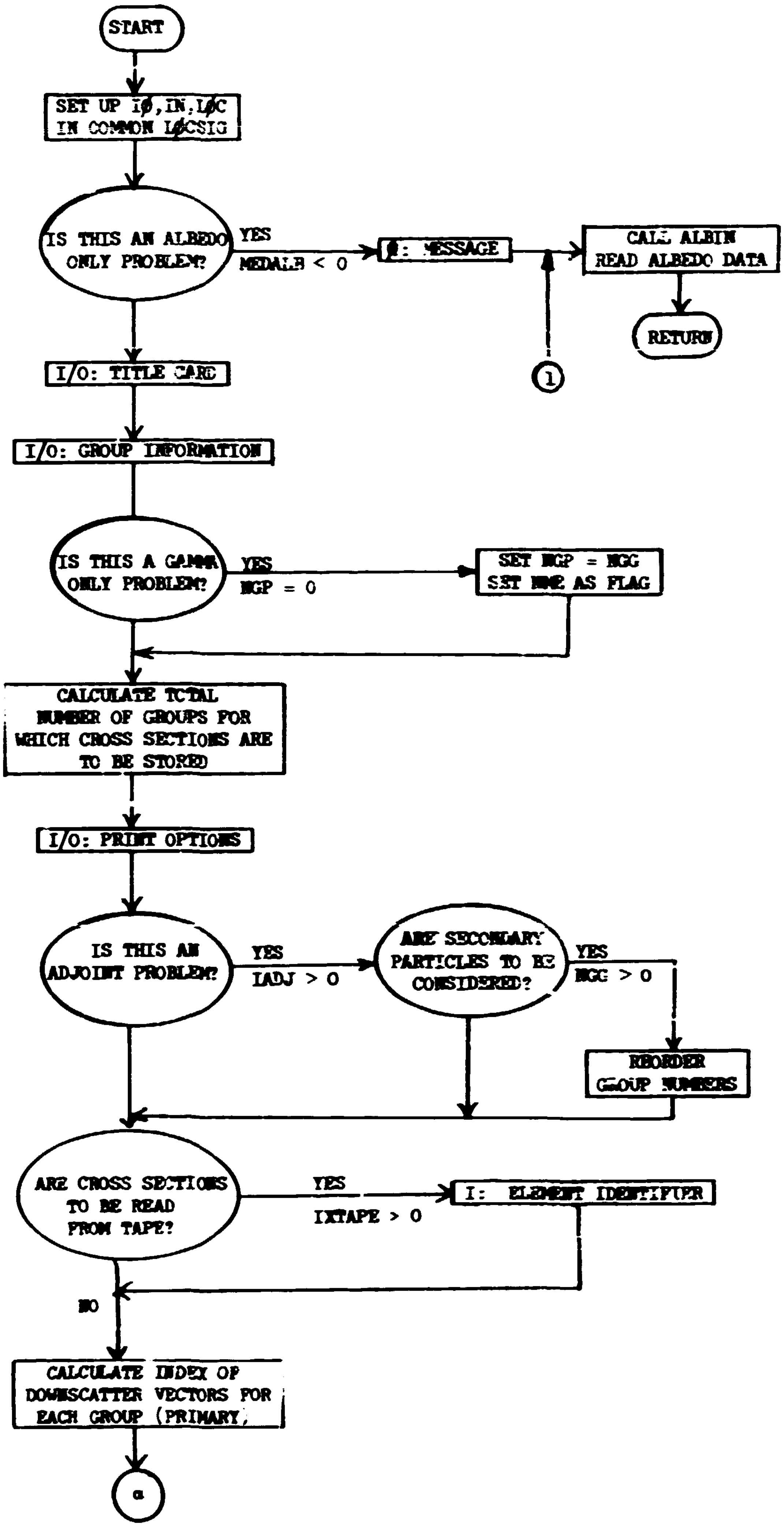
MEDALB - set to 7777 if there is no albedo surface in problem,

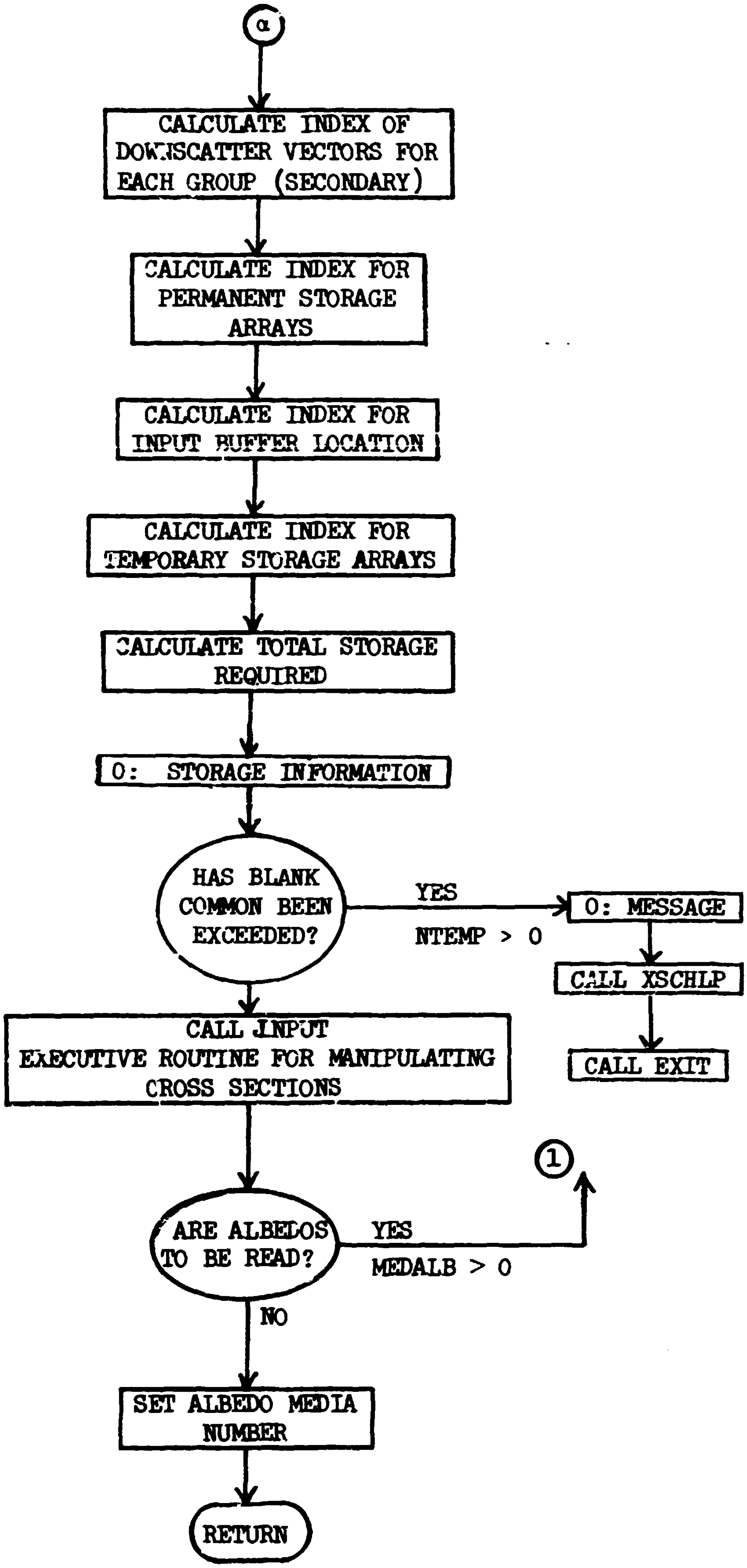
NLAST - the last cell of permanent storage required,

Significant internal variables:

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

Subroutine XSEC


START
SET UP IØ,IN,LØC
IN COMMON LØCSIG
IS THIS AN ALBEDO ONLY PROBLEM?
YES
NEDALB < 0
Ø: MESSAGE
CALL ALBIN
READ ALBEDO DATA
RETURN
1
I/O: TITLE CARD
I/O: GROUP INFORMATION
IS THIS A GAMMA ONLY PROBLEM?
YES
NGP = 0
SET NGP = NGG
SET NNE AS FLAG
CALCULATE TOTAL NUMBER OF GROUPS FOR WHICH CROSS SECTIONS ARE TO BE STORED
I/O: PRINT OPTIONS
IS THIS AN ADJOINT PROBLEM?
YES
IADJ > 0
ARE SECONDARY PARTICLES TO BE CONSIDERED?
YES
NGG > 0
REORDER GROUP NUMBERS
ARE CROSS SECTIONS TO BE READ FROM TAPE?
YES
IXTAPE > 0
I: ELEMENT IDENTIFIER
NO
CALCULATE INDEX OF DOWNSCATTER VECTORS FOR EACH GROUP (PRIMARY)
α


α
CALCULATE INDEX OF DOWNSCATTER VECTORS FOR EACH GROUP (SECONDARY)
CALCULATE INDEX FOR PERMANENT STORAGE ARRAYS
CALCULATE INDEX FOR INPUT BUFFER LOCATION
CALCULATE INDEX FOR TEMPORARY STORAGE ARRAYS
CALCULATE TOTAL STORAGE REQUIRED
0: STORAGE INFORMATION
HAS BLANK COMMON BEEN EXCEEDED?
YES
NTEMP > 0
0: MESSAGE
CALL XSCHLP
CALL EXIT
CALL INPUT EXECUTIVE ROUTINE FOR MANIPULATING CROSS SECTIONS
1
ARE ALBEDOS TO BE READ?
YES
MEDALB > 0
NO
SET ALBEDO MEDIA NUMBER
RETURN

IV. Diagnostic Module

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 this module 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.

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.

*HELPER is a slight revision of TDUMP.[12]

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,

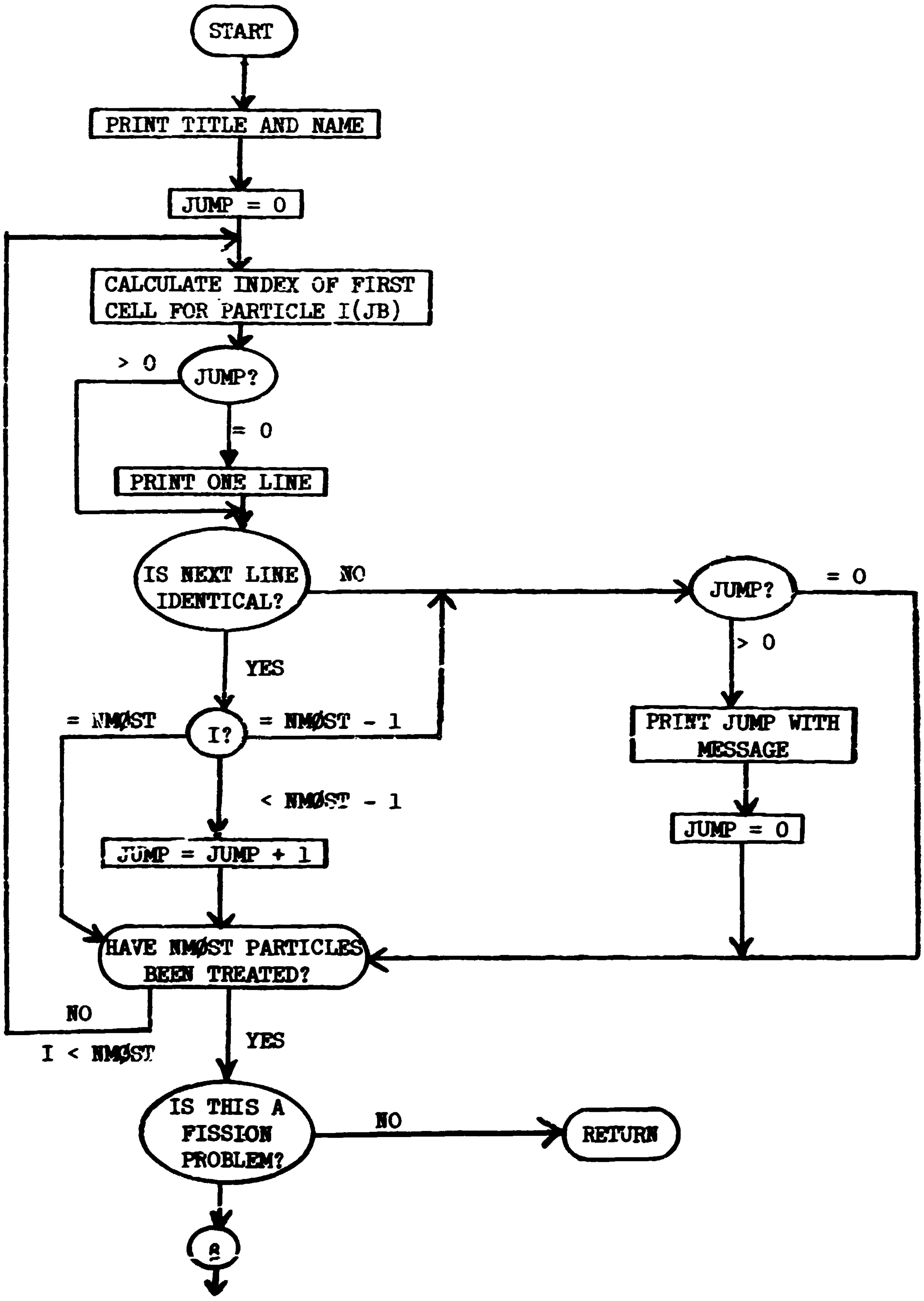
NMØST - maximum number of particles allowed for in the bank(s),

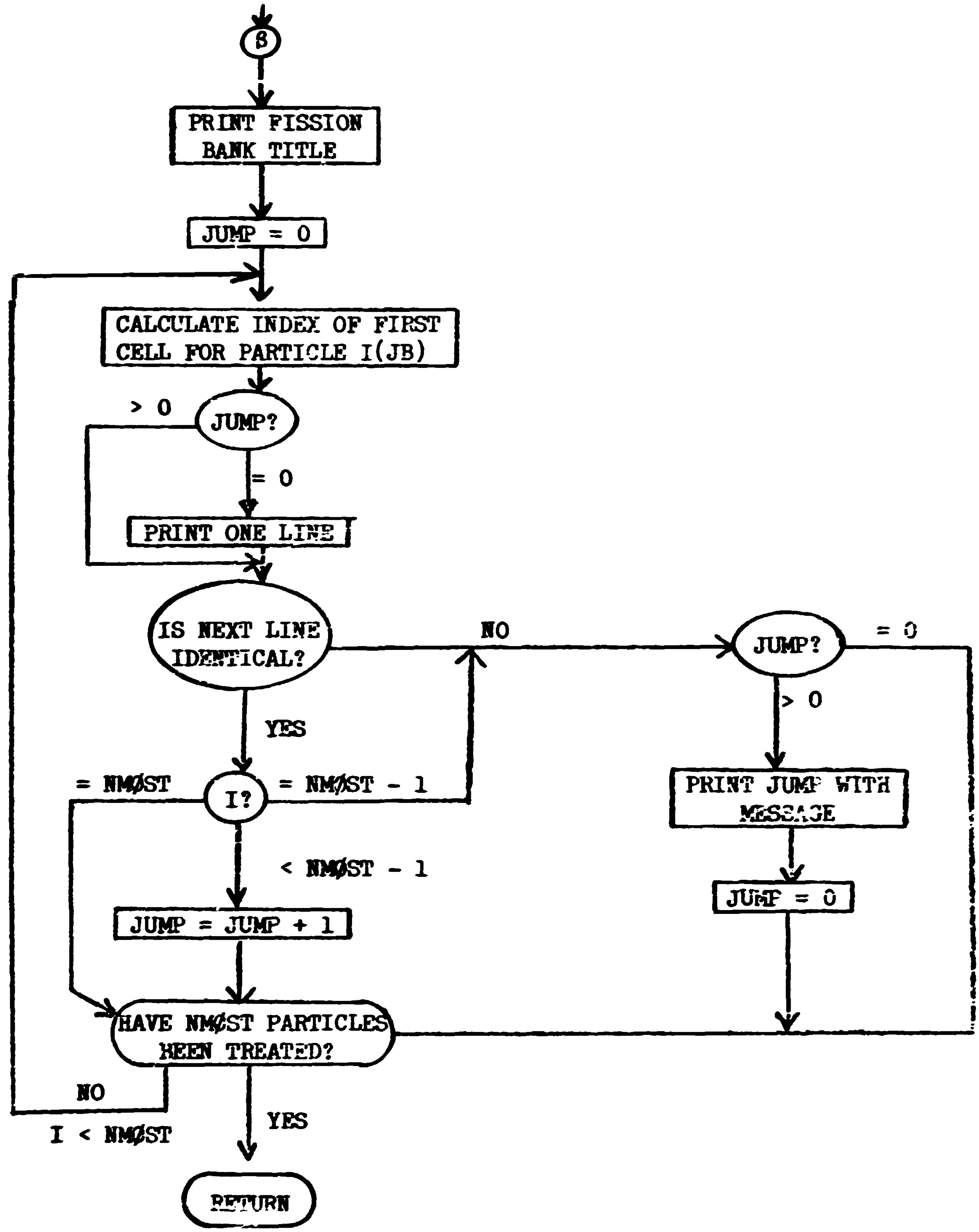
IO - logical unit for output,

MFISTP - index indicating that fissions are to be considered if > 0,

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

Subroutine BNKHLP (NAME)




β
PRINT FISSION BANK TITLE
JUMP = 0
CALCULATE INDEX OF FIRST CELL FOR PARTICLE I(JB)
> 0
JUMP?
= 0
PRINT ONE LINE
IS NEXT LINE IDENTICAL?
NO
JUMP?
= 0
> 0
YES
= NMØST
I?
= NMØST - 1
PRINT JUMP WITH MESSAGE
< NMØST - 1
JUMP = 0
JUMP = JUMP + 1
HAVE NMØST PARTICLES BEEN TREATED?
NO
I < NMØST
YES
RETURN

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 just after each problem is completed, and also 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.

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,

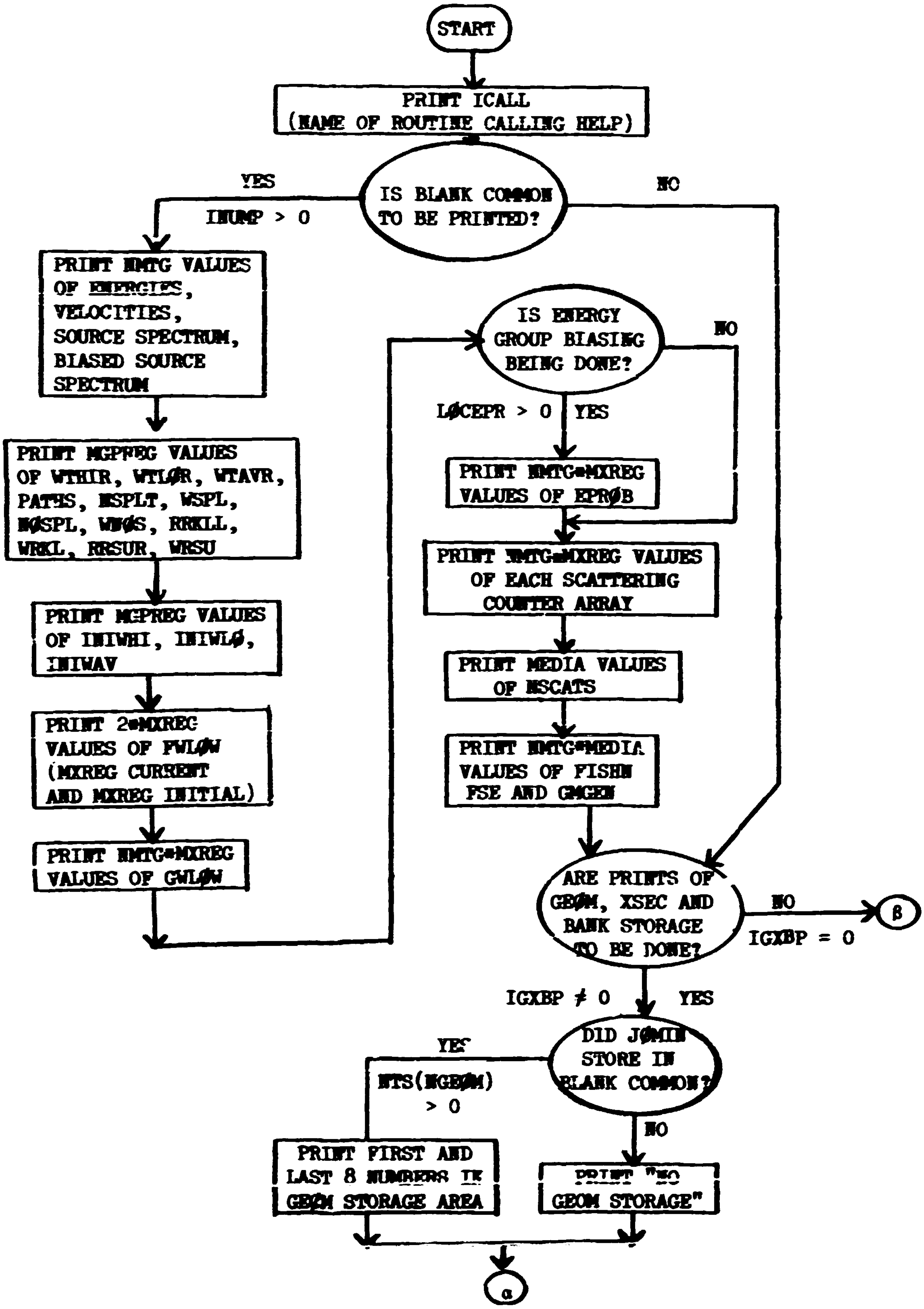
NGEØM - location of cell one of geometry data storage,

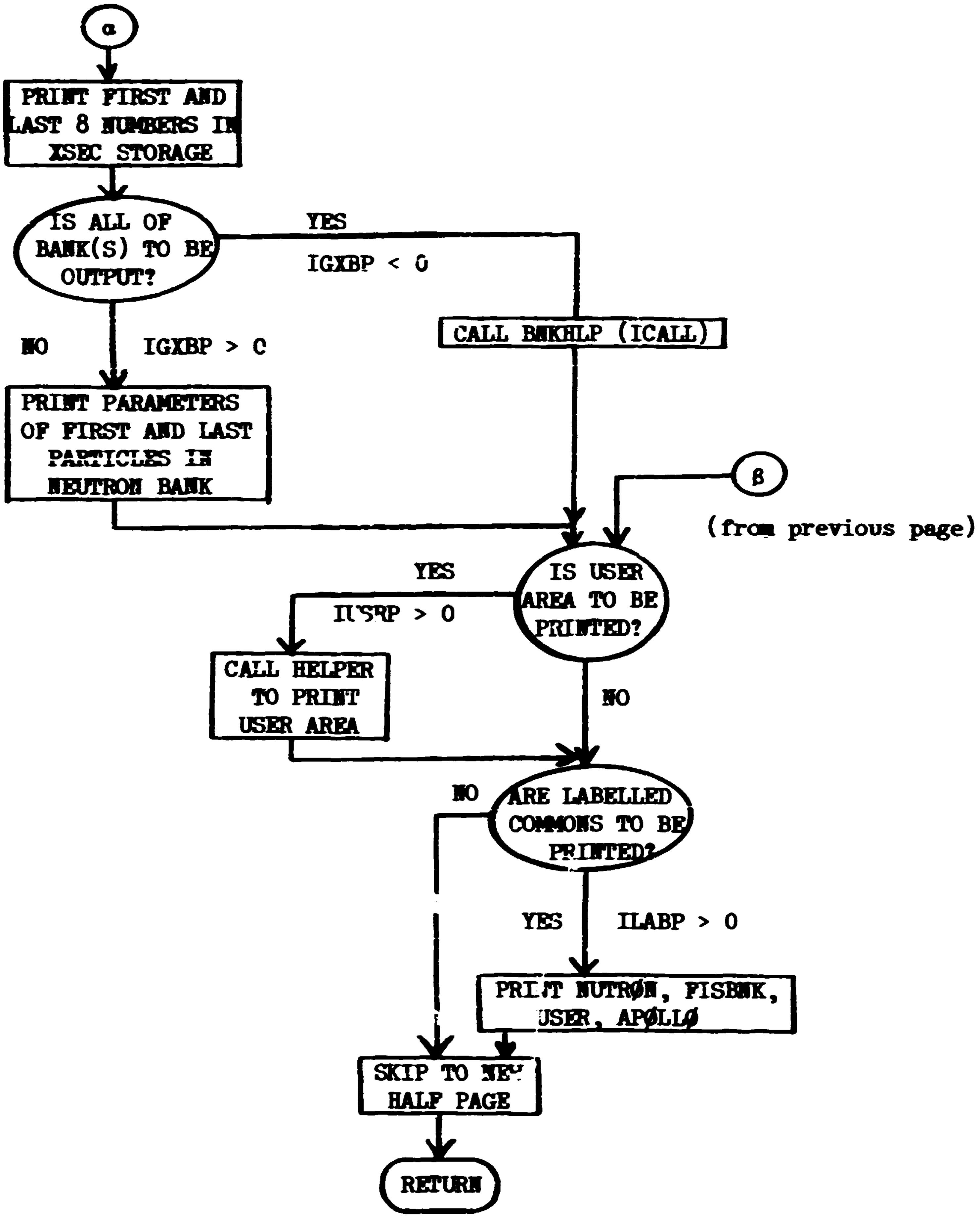
NSIGL - location of last cell in permanent cross-section storage,

NLAST - last cell used by neutron or fission bank,

NLEFT - number of cells available to user beyond banks.

Subroutine HELP (ICALL,INUMP,ILABP,IGXBP,IUSRP)


START
PRINT ICALL (NAME OF ROUTINE CALLING HELP)
IS BLANK COMMON TO BE PRINTED?
YES
INUMP > 0
NO
PRINT NMTG VALUES OF ENERGIES, VELOCITIES, SOURCE SPECTRUM, BIASED SOURCE SPECTRUM
PRINT MGPREG VALUES OF WTHIR, WTLØR, WTAVR, PATHS, NSPLT, WSPL, NØSPL, WNØS, RRKLL, WRKL, RRSUR, WRSU
PRINT MGPREG VALUES OF INIWHI, INIWLØ, INIWAV
PRINT 2*MXREG VALUES OF FWLØW (MXREG CURRENT AND MXREG INITIAL)
PRINT NMTG*MXREG VALUES OF GWLØW
IS ENERGY GROUP BIASING BEING DONE?
NO
LØCEPR > 0
YES
PRINT NMTG*MXREG VALUES OF EPRØB
PRINT NMTG*MXREG VALUES OF EACH SCATTERING COUNTER ARRAY
PRINT MEDIA VALUES OF NSCATS
PRINT NMTG*MEDIA VALUES OF FISHN FSE AND GMGEN
ARE PRINTS OF GEØM, XSEC AND BANK STORAGE TO BE DONE?
NO
IGXBP = 0
β
IGXBP ≠ 0
YES
DID JØMIN STORE IN BLANK COMMON?
YES
NTS(NGEØM) > 0
NO
PRINT FIRST AND LAST 8 NUMBERS IN GEØM STORAGE AREA
PRINT "NO GEOM STORAGE"
α


α
PRINT FIRST AND LAST 8 NUMBERS IN XSEC STORAGE
IS ALL OF BANK(S) TO BE OUTPUT?
YES
IGXBP < 0
CALL BNKHLP (ICALL)
NO
IGXBP > 0
PRINT PARAMETERS OF FIRST AND LAST PARTICLES IN NEUTRON BANK
β
(from previous page)
YES
IUSRP > 0
IS USER AREA TO BE PRINTED?
CALL HELPER TO PRINT USER AREA
NO
NO
ARE LABELLED COMMONS TO BE PRINTED?
YES
ILABP > 0
PRINT NUTRØN, FISBNK, USER, APØLLØ
SKIP TO NEW HALF PAGE
RETURN

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,

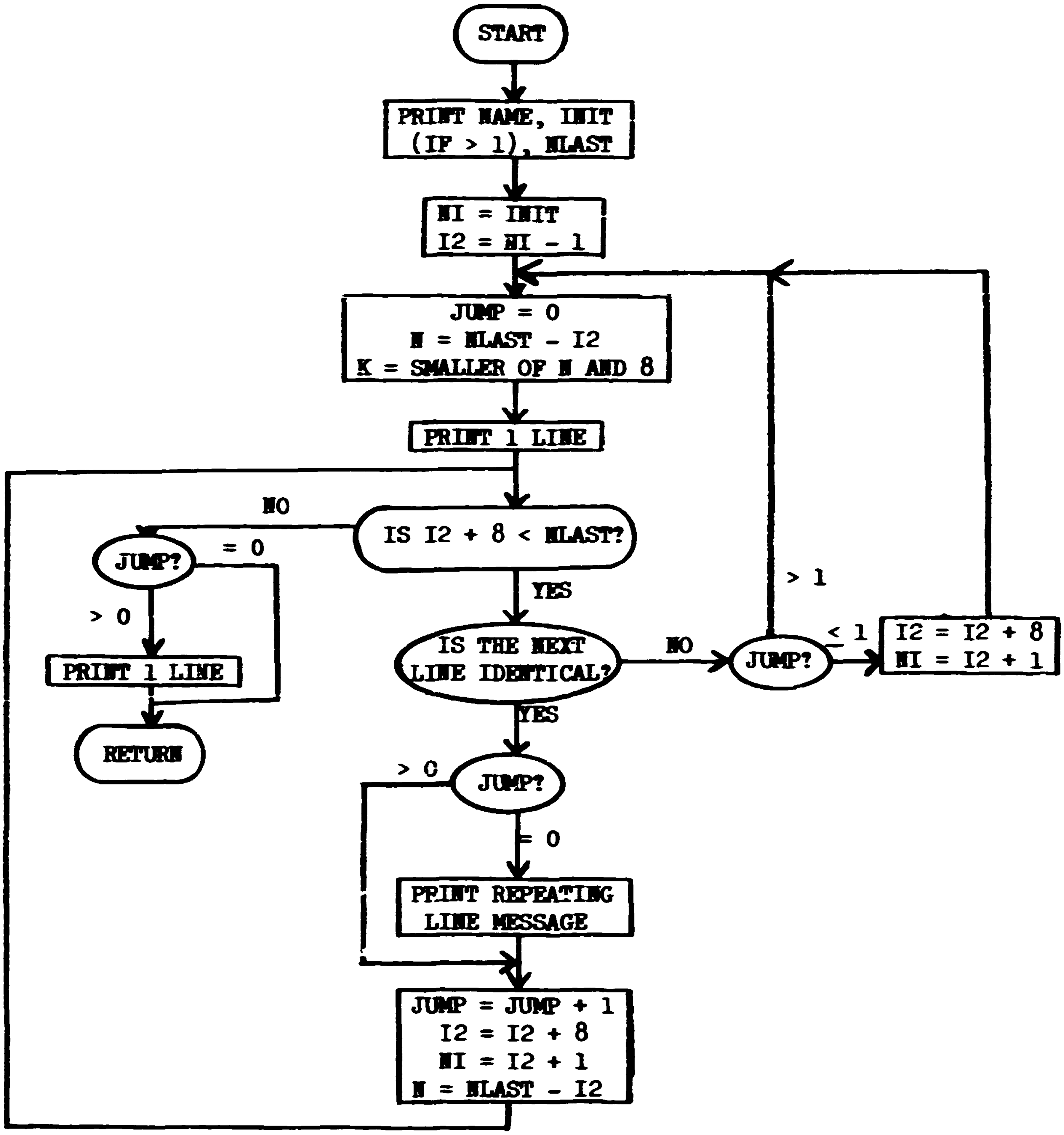
INIT - first 4-byte word of array A to be output,

NLAST - last 4-byte word of array to be output,

NAME - 4 hollerith characters to be used as a label,

IO - output unit.

Subroutine HELPER (A,INIT,NLAST,NAME,IO)



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 ORNL; 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.

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 VI for layout of the cross-section area.) This subroutine may be called from any location.

Called from: READSG, PTHETA, XSEC, and ANGLE (just before error calls).

Subroutines called: HELPER

Functions used: LØC

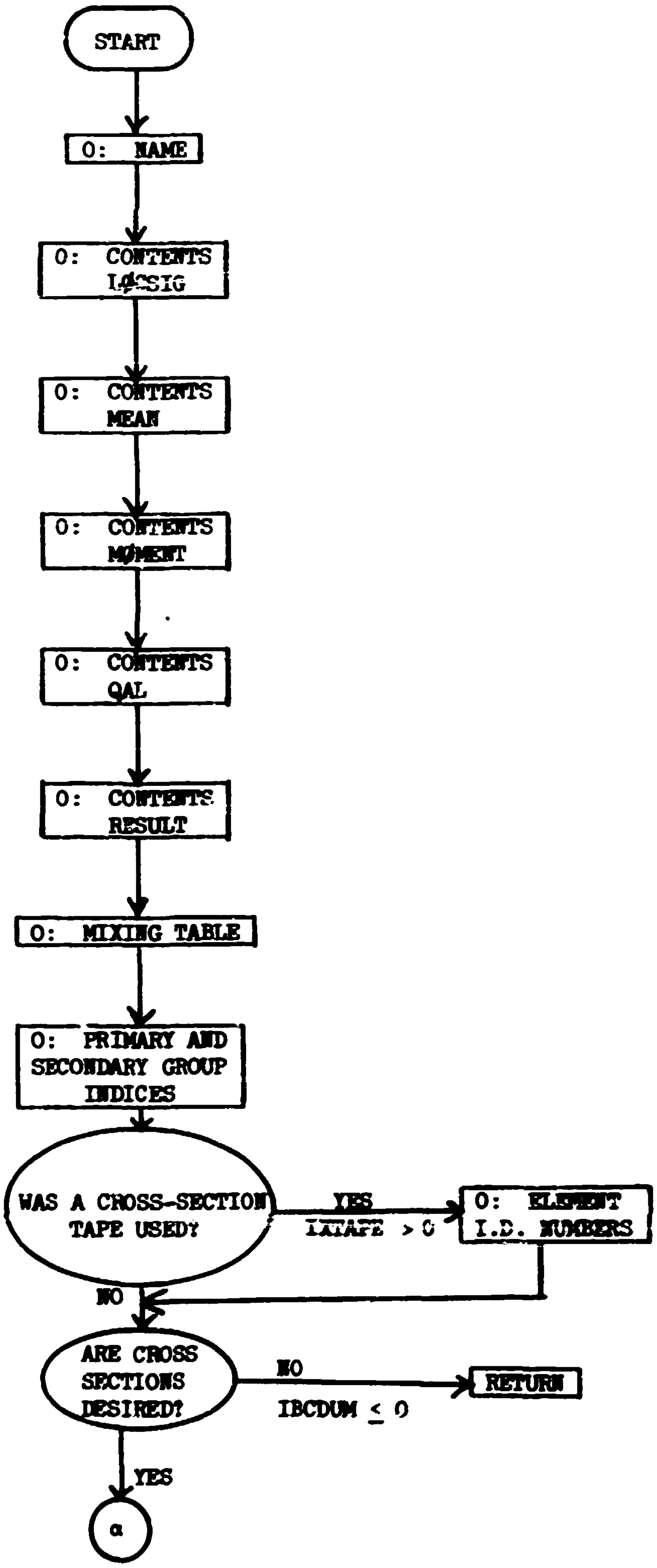
Commons required: Blank, LØCSIG, MEANS, MØMENT, QAL, RESULT.

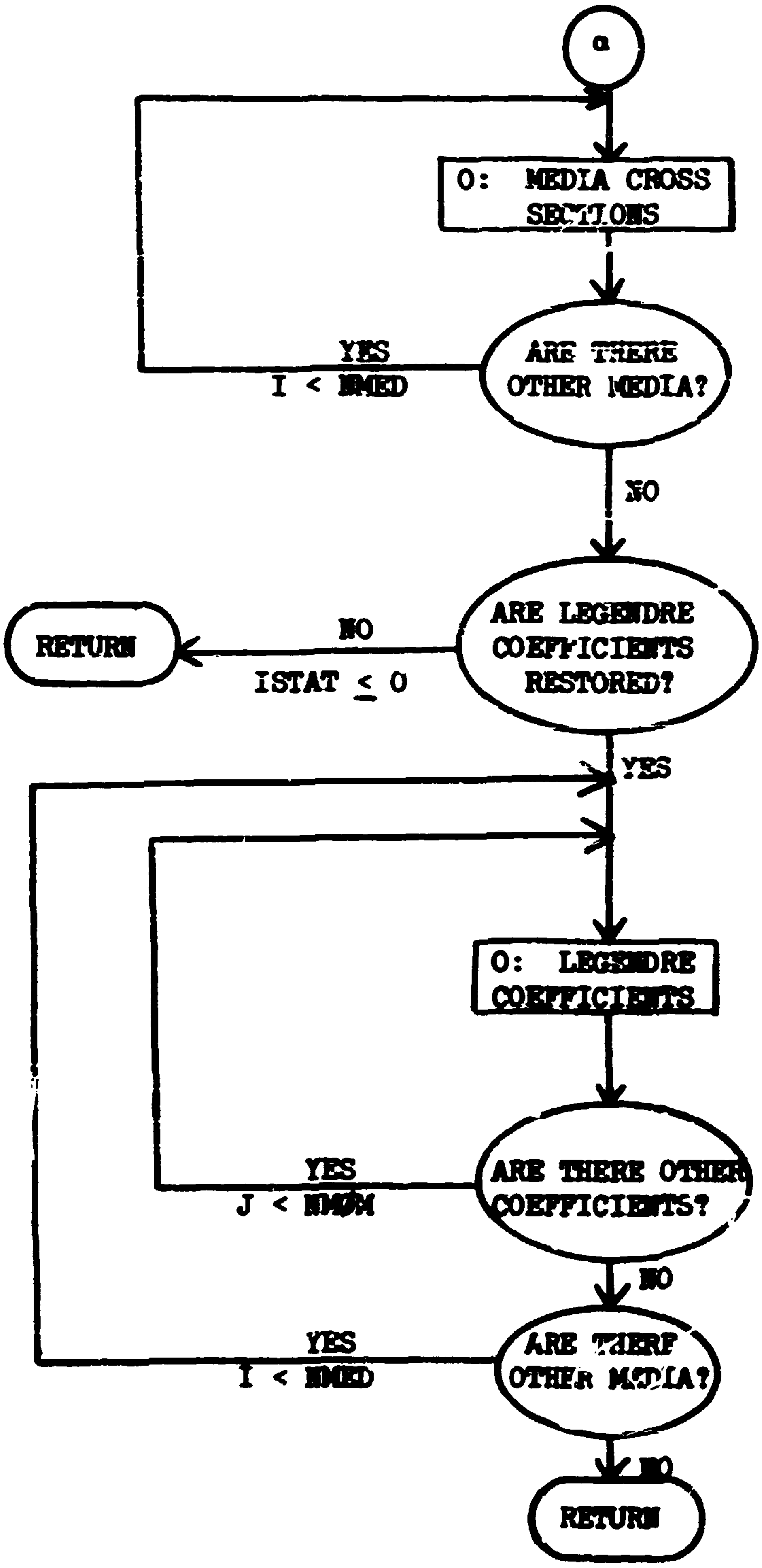
Variables required:

IBCDUM - contents of blank common are printed if > 0.

NAME - a four-character word to indicate the calling program.

Subroutine XSCHLP (IBCDUM, NAME)




a
0: MEDIA CROSS SECTIONS
ARE THERE OTHER MEDIA?
YES
I < NMED
NO
ARE LEGENDRE COEFFICIENTS RESTORED?
NO
ISTAT ≤ 0
RETURN
YES
0: LEGENDRE COEFFICIENTS
ARE THERE OTHER COEFFICIENTS?
YES
J < NMØM
NO
ARE THERE OTHER MEDIA?
YES
I < NMED
NO
RETURN

V. Analysis Interface and Sample User Routines

The MORSE interface to user-provided analysis routines is through calls to function DIREC, and subroutines GTMED, SCØRIN, SØURCE, and especially BANKR. Function DIREC supplies the dot product between the neutron direction vector and the most important direction. It is used by GETETA, which determines the length of the next flight, to vary the amount of path-length biasing depending on whether the particle is traveling in an important direction or not. If path-length biasing is not desired (or if it is desired to bias all paths independently of direction), DIREC should return 1.0. GTMED is used to relate cross-section and geometry media. It is called from every routine needing cross-section data. These calling routines have available the medium of the point of interest, as specified by the geometry data, and need the proper medium to give the cross-section routines. In most cases, the geometry and cross-section media are the same, but for special cases such as the infinite homogeneous media with a boundary crossing estimator two different media for GEØM are required. All data required by user-written routines are input by subroutine SCØRIN which is called after the general problem specification, geometry, and cross-section data are input. Subroutine SØURCE is called for each source particle (including neutrons just produced by fission) so that the user may specify the phase space coordinates of each (if it is not desirable to use the constant values specified by input cards to the walk routines).

BANKR is the primary interface to the analysis package, being called with as many as 17 values of the argument index to direct the analysis. These arguments and their meaning are outlined in Table VII.

It should be noted that not all the BANKR calls listed in Table VII are actually programmed in the code (those not programmed are included as comments); the user may have to add these calls for his special purposes. Several labelled commons transfer data for use in the analysis, and, in addition, the unused portion of blank common is made available. Data which are determined by the problem specification (which are not modified by the walk or whose initial value may be useful) are loaded in common USER by subroutine INPUT. These quantities are given in Table VIII.

Table VII. 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	After 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.

Table VIII. Definition of Variables in Common USER

AGSTRT	Initial age (input on card D).
WTSTRT	Initial weight (input on card C).
XSTRT	Initial x position (input on card D).
YSTRT	Initial y position (input on card D).
ZSTRT	Initial z position (input on card D).
DFF	Normalization for adjoint problems - calculated in SØRIN.
EBØTN	Lower energy boundary of last neutron group.
EBØTG	Lower energy boundary of last gamma-ray group.
TCUT	Age limit (input on card C).
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 flow chart for subroutine INPUT.
NGPQTG	Lowest energy gamma-ray group.
NGPQTN	Lowest energy neutron group.
NITS	Number of batches (input on card B).
NLAST	Last cell in blank common used by either cross-section package or bank(s), whichever is larger.
NLEFT	Number of cells in blank common available to user.
NMGP	Number of primary (neutron) groups (input on card B).
NMTG	Number of total groups (input on card B).
NSTRT	Number of source particles for each batch (input on card B).

The user will also need common NUTRØN which contains prior and present collision parameters (the pre-collision weight is also provided).

All other variables in the walk which may be needed by the user should be transmitted by the primary interface routine, BANKR, as arguments in the called routine. See BANKR writeup for examples.

Sample User Routines

The problem chosen for this example is to calculate fluence at up to 20 distances from a point, isotropic source in an infinite medium. A boundary-crossing estimator is used along with alternating geometry media 1 and 2 in concentric spherical shells. The information required by the sample analysis routines is passed by common DET. The variables required are defined in Table IX. Descriptions of each routine including flow charts and listings follow.

A description of the versatile analysis package SAMBO is contained in reference 1. Some of the user routines described here are replaced by more general routines in SAMBO; other routines complement those in SAMBO.

Table IX. Definition of Variables in Common DET

Variable	Definition
ND	Number of detectors.
NSCAPE	Counter for boundary crossings beyond the last detector.
RAD(20)	Radii in cm of the spherical detectors (must be media boundaries).
NN(20)	Number of estimates at each detector.
UD(20)	Uncollided response for the current batch.
SUD(20)	Uncollided response (UD) summed over batches.
SUD2(20)	Sum, over batches, of the squares of the uncollided response estimates weighted with NSØRC* (sum of UD**2/NSØRC).
SD(20)	Total (uncollided plus collided) response for the current batch.
SSD(20)	Total response (SD) summed over batches.
SSD2(20)	Sum, over batches, of the squares of the total response estimates weighted with NSØRC (sum of SD**2/NSØRC).
FDCF(100)	Response function array.

*NSØRC is the number of particles starting the batch.

Subroutine BANKR (NBNKID)

The function of BANKR is to call analysis and diagnostic subroutines as specified by the user. The particular subroutines called in the analysis module are determined by the index NBNKID. In this problem: BANKR (-4) calls NRUN; BANKR (-3) calls NBATCH; BANKR (-2) calls STBTCH; BANKR (-1) calls STRUN and HELP; BANKR (1) calls SDATA; and BANKR (7) calls BDRYX. Any other values of NBNKID result in a return.

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, SDATA, BDRYX, HELP.

Commons required: APØLLØ.

Variables required:

NBNKID - an index which identifies the type of collision and/or subroutine called (NBNKID = -4, -3, -2, -1, 1, 2, ... 13),

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),

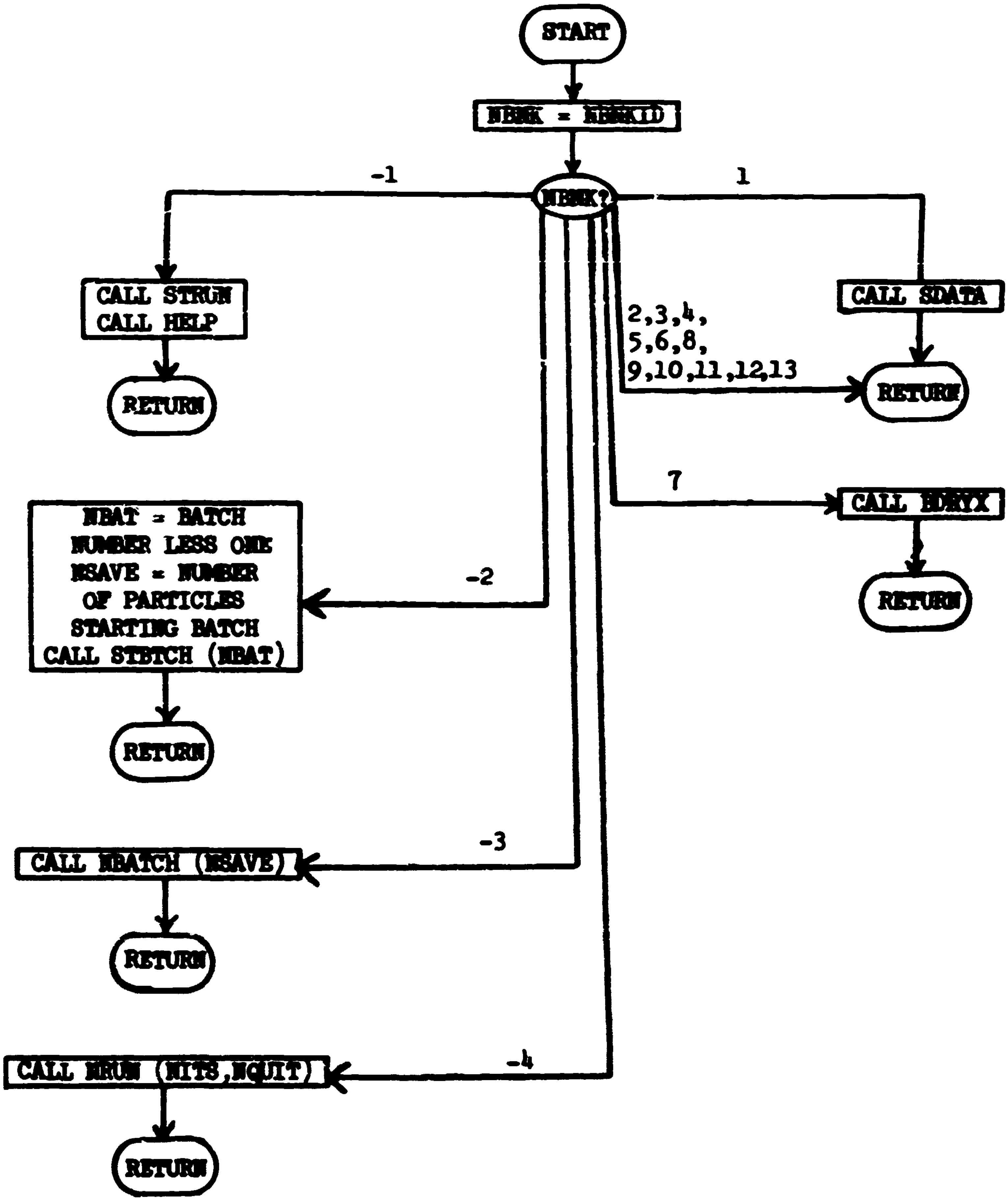
NMEM - number of particles which remain to be processed in a given batch.

Significant internal variables:

NBAT - the batch number less one,

NSAVE - the number of particles starting the current batch.

Subroutine BANKR (NBNKID)



```
      SUBROUTINE BANKR(NBNKID)                                          BANKR 10
C  DO NOT CALL EUCLID FROM BANKR(7)                                     BANKR 20
      COMMON /APOLLO/ AGSTRT,DDF,DEADWT(5),ETA,ETATH,ETAUSD,UINP,VINP,  BANKR 30
     1 WINP,WTSTRT,XSTRT,YSTRT,ZSTRT,TCUT,XTRA(10),                     BANKR 31
     2   IO,II,MEDIA,IADJM,ISBIAS,ISOUR,ITERS,ITIME,ITSTR,LOCWTS,LOCFWL,BANKR 32
     3 LOCEPR,LOCNSC,LOCFSN,MAXGP,MAXTIM,MEDALB,MGPREG,MXREG,NALB,      BANKR 33
     4 NDEAD(5),NEWNM,NGEOM,NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,    BANKR 34
     5 NKCALC,NKILL,NLAST,NMEM,NMGP,NMOST,NMTG,NOLEAK,NORMF,NPAST,      BANKR 35
     6 NPSCL(13),NQUIT,NSIGL,NSOUR,NSPLT,NSTRT,NXTRA(10)                BANKR 36
      COMMON /NUTRON/ NAME,NAMEX,IG,IGO,NMED,MEDOLD,NREG,U,V,W,UOLD,VOLDBANKR 40
     1 ,WOLD,X,Y,Z,XOLD,YOLD,ZOLD,WATE,OLDWT,WTBC,BLZNT,BLZON,AGE,OLDAGEBANKR 41
      NBNK = NBNKID                                                     BANKR 50
      IF (NBNK) 100,103,140                                             BANKR 60
  100 NBNK = NBNK + 5                                                   BANKR 70
      GO TO (104,103,102,101),NBNK                                      BANKR 80
  101 CALL STRUN                                                        BANKR 90
C     CALL HELP(4HSTRU,1,1,1,1)                                         BANK 100
      RETURN                                                            BANK 110
  102 NBAT = NITS - ITERS                                               BANK 120
      NSAVE = NMEM                                                      BANK 130
      CALL STBTCH(NBAT)                                                 BANK 140
C   NBAT IS THE BATCH NO. LESS ONE                                      BANK 150
      RETURN                                                            BANK 160
  103 CALL NBATCH(NSAVE)                                                BANK 170
C   NSAVE IS THE NO. OF PARTICLES STARTED IN THE LAST BATCH             BANK 180
      RETURN                                                            BANK 190
  104 CALL NRUN(NITS,NQUIT)                                             BANK 200
C   NITS IS THE NO. OF BATCHES COMPLETED IN THE RUN JUST COMPLETED      BANK 210
C   NQUIT .GT. 1 IF MORE RUNS REMAIN                                    BANK 220
C         .EQ. 1 IF THE LAST SCHEDULED RUN HAS BEEN COMPLETED           BANK 230
C         IS THE NEGATIVE OF THE NO. OF COMPLETE RUNS, WHEN AN          BANK 240
C                EXECUTION TIME KILL OCCURS                             BANK 250
      RETURN                                                            BANK 260
  140 GO TO (1,2,3,4,5,6,7,8,9,10,11,12),NBNK                           BANK 270

C  NBNKID   COLL TYPE   BANKR CALL      NBNKID   COLL TYPE   BANKR CALL BANK 280
C     1      SOURCE     YES (NSOUR)       2       SPLIT      NO  (TESTW)BANK 290
C     3      FISSION    YES (FPROB)       4       GAMGEN     NO  (GSTOREBANK 300
C     5      REAL COLL  YES (NORSE)       6       ALBEDO     YES (NORSE)BANK 310
C     7      BDRYX      YES (NXTCOL)      8       ESCAPE     YES (NXTCOLBANK 320
C     9      E-CUT      NO  (NORSE)      10       TIME KILL  NO  (NORSE)BANK 330
C    11      R R KILL   NO  (TESTW)      12       R R SURV   NO  (TESTW)BANK 340
C    13      GAMLOST    NO  (GSTORE)                                    BANK 350
    1 CALL SDATA                                                        BANK 360
      RETURN                                                            BANK 370
    2 RETURN                                                            BANK 380
    3 RETURN                                                            BANK 390
    4 RETURN                                                            BANK 400
    5 RETURN                                                            BANK 410
    6 RETURN                                                            BANK 420
    7 CALL BDRYX                                                        BANK 430
      RETURN                                                            BANK 440
    8 RETURN                                                            BANK 450
    9 RETURN                                                            BANK 460
   10 RETURN                                                            BANK 470
   11 RETURN                                                            BANK 480
   12 RETURN                                                            BANK 490
      END                                                               BANK 500
```

Subroutine BDRYX

This routine is called whenever the particle in the walk encounters a change in geometry media. If the source-to-collision distance corresponds to a detector position, the reciprocal of the cosine of the angle from the radius vector is used as a fluence estimate. The response value for the appropriate energy group modifies the estimate, which is then stored in the counter for the appropriate detector.

Called from: BANKR (7)

Subroutines Called: ERRØR (library)
ABS (library function)

Commons required: USER, NUTRØN, DET

Variables required:

X, Y, Z, U, V, W, WATE (from common NUTRØN, see page 12)

ND, NSCAPE, NN(I), FDCF(I), SD(I), RAD(I) (from common DET, see page 169)

Variables changed: NSCAPE, NN, SD.

Significant internal variables:

R21 - radial distance to boundary crossing,

R2 - 99% of R21,

R22 - 101% of R21,

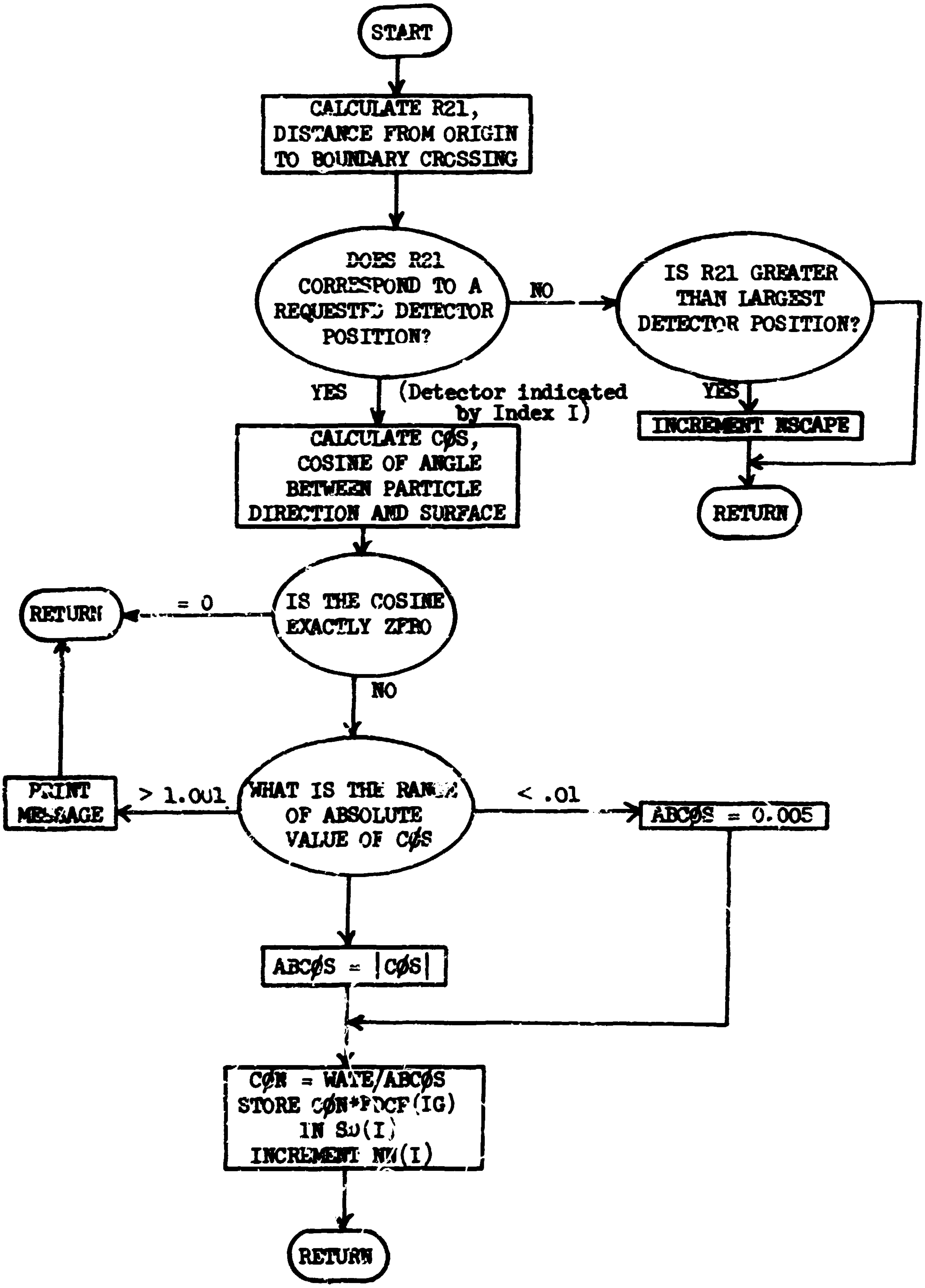
CØS - cosine of angle between particle direction and radius vector,

ABCØS - absolute value of CØS,

CØN - fluence estimate,

CØND - response estimate.

Subroutine BDRYX



```
      SUBROUTINE BDRYX
C
C     FOR USE IN SPHERICAL GEOMETRY ONLY
C
C     IDENTIFIES DETECTOR POSITION WITH A BOUNDARY CROSSING AND THEN
C       CALCULATES AND SUMS QUANTITIES OF INTEREST FOR EACH BATCH.
C
      COMMON /USER/ AGSTRT,WTSTRT,XSTRT,YSTRT,ZSTRT,EFF,EBOTN,EBOTG,
     1 TCUT,IO,I1,IADJM,NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,NLAST,
     2 NLEFT,NMGP,NMTG,NSTRT
      COMMON /DET/ ND,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      COMMON /NUTRON/ NAME,NAMEX,IG,IGC,NMED,MEDOLD,NREG,U,V,W,UOLD,VOLD
     1 ,WOLD,X,Y,Z,XOLD,YOLD,ZOLD,WATE,OLDWT,WTBC,BLZNT,BLZON,AGE,OLDAGE
      R21 = SQRT (X**2 + Y**2 + Z**2)
      R2 = R21*0.99
      R22 = R21*1.01
      DO 5 I=1,ND
      IF (R2-RAD(I)) 15,15,5
 5    CONTINUE
      NSCAPE=NSCAPE+1
 10   RETURN
 15   IF (R22-RAD(I)) 10,20,20
 20   ERA = U*X + V*Y + W*Z
      COS = ERA/R21 - 1.E-10
      IF (COS)  30,25,30
 25   WRITE (IO,1000)
 1000 FORMAT(1H0,14H COS=0.,RETURN)
      RETURN
 30   ABCOS=ABS (COS)
      IF (ABCOS-1.0001) 40,40,35
 35   WRITE (IO,1010) ABCOS
 1010 FORMAT(1H0,'ABCOS.GT.1. = 'E10.4)
      CALL ERROR
 40   IF (ABCOS-0.01) 45,50,50
 45   ABCOS = 0.005
 50   CON=WATE/ABCOS
      NN(I) = NN(I) + 1
      COND = CON*FDCF(IG)
      SD(I) = SD(I) + COND
      RETURN
      END
```

Function DIREC (DUMMY)

This function provides the dot product of the neutron direction vector and the radius vector. Thus DIREC = 1.0 for an outgoing neutron and = -1.0 for an inward going neutron. These values result in maximum path stretching and shrinking, respectively, when used in the calling routine GETETA.

Called from: GETETA

Function used: SQRT (library)

Commons required: NUTRØN

Variables required:

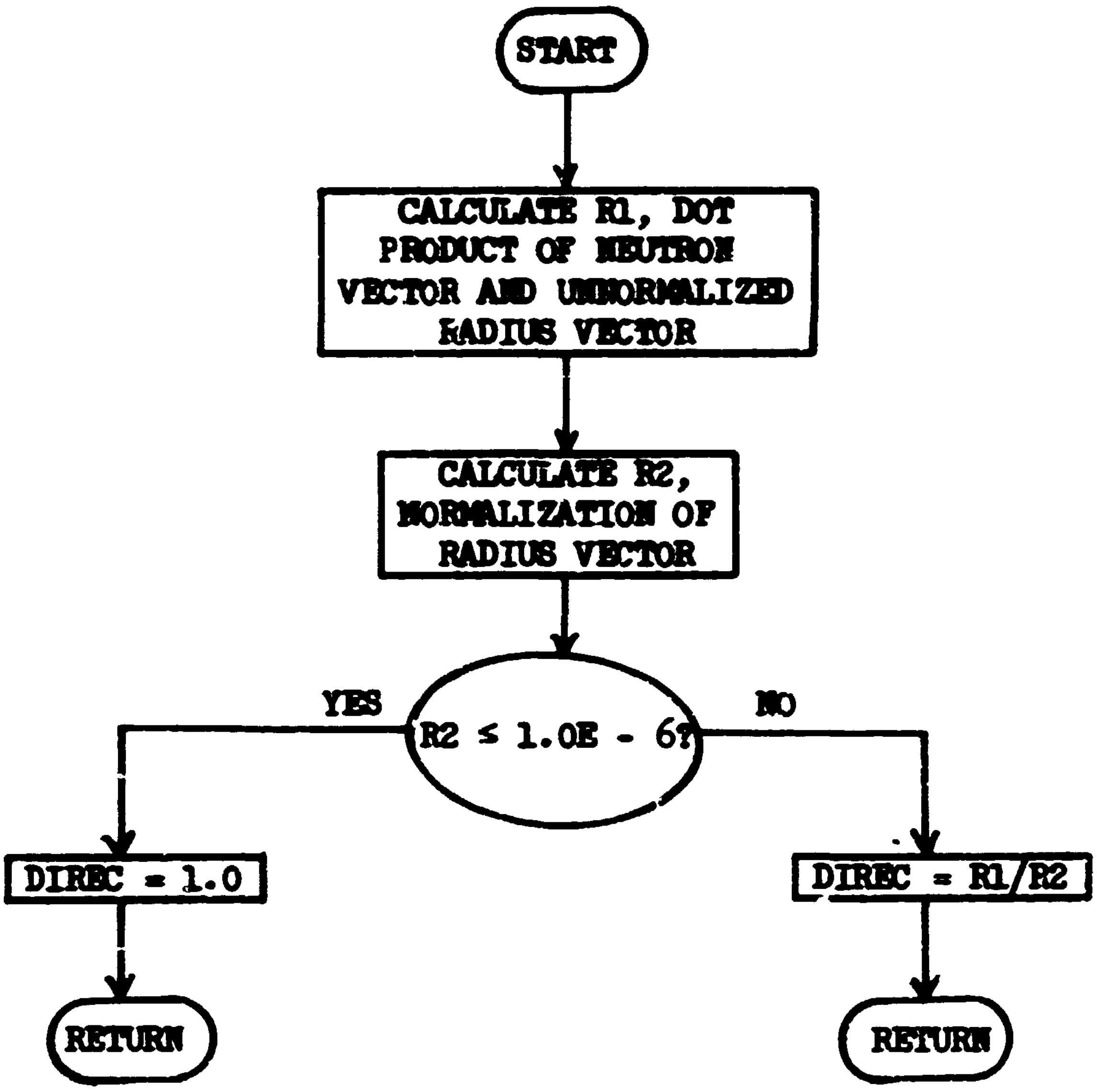
UØLD, VØLD, WØLD - prior collision direction cosines (at this point they are equal to the current collision values),

XØLD, YØLD, ZØLD - coordinates of prior collision site (at this point they are equal to the current collision values).

Variables changed:

DIREC - the function value.

Function DIREC (DUMMY)


START
CALCULATE R1, DOT PRODUCT OF NEUTRON VECTOR AND UNNORMALIZED RADIUS VECTOR
CALCULATE R2, NORMALIZATION OF RADIUS VECTOR
R2 ≤ 1.0E - 6?
YES
NO
DIREC = 1.0
DIREC = R1/R2
RETURN
RETURN

```
      FUNCTION DIREC(F)
C
C     SPHERICAL GEOMETRY VERSION
C
      COMMON /NUTRON/ NAME,NAMEX,IG,IGO,NMED,MEDOLD,NREG,U,V,W,UOLD,VOLD
     1 ,WOLD,X,Y,Z,XOLD,YOLD,ZOLD,WATE,OLPNT,WTBC,BLTNT,BLZON,AGE,OLDAGE
      R1=UOLD*XOLD+VOLD*YOLD+WOLD*ZOLD
      R2=SQRT (XOLD**2+YOLD**2+ZOLD**2)
      IF (R2 - 1.E-6) 10,10,5
 5    COS=R1/R2
      DIREC=COS
      RETURN
 10   DIREC=1.
      RETURN
      END
```

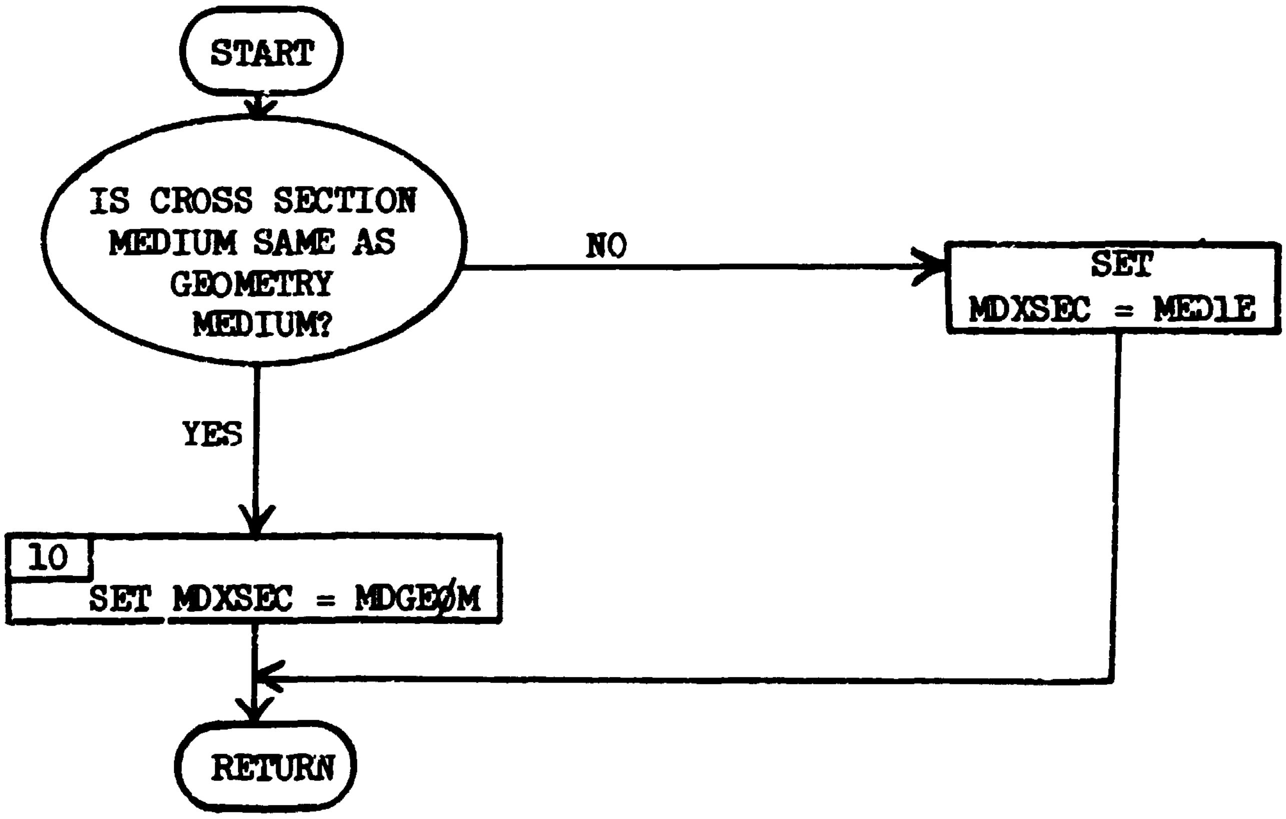
Subroutine GTMED (MDGEØM, MDXSEC)

This subroutine allows one to equate the cross sections for two different geometric media. Thus, if one uses a boundary crossing estimator, GEØM requires that the media on both sides of the boundary differ. However, for a homogeneous problem, the transport needs only one cross section to be stored. For any problem not involving a boundary crossing estimator for a homogeneous system, MDGEØM and MDXSEC may be equivalenced and the subroutine calls removed.

A data statement sets the two media numbers that are to have the same cross sections.

Called from: MØRSE, FPRØB, NSIGTA, CØLISN, PTHETA, FISGEN, GAMGEN
Variables required: MDGEØM, MED1E, MED2E
Variables changed: MDXSEC

Subroutine GTMED (MDGEØM, MDXSEC)



```
      SUBROUTINE GTMED(MDGEOM,MDXSEC)                                   GTMED 10
      DATA MED1E/1/,MED2E/2/                                            GTMED 20
      IF (MDGEOM - MED2E) 10,5,10                                       GTMED 30
5     MDXSEC = MED1E                                                    GTMED 40
      RETURN                                                            GTMED 50
10    MDXSEC = MDGEOM                                                   GTMED 60
      RETURN                                                            GTMED 70
      END                                                               GTMED 80
```

Subroutine NBATCH (NSØRC)

This routine is called at the end of each batch to perform the sums needed for calculation of batch statistics. Provision is made, although not used in this case, for batches of different sizes. Because of this, the summation of the square of the accumulated estimate is divided by the number of particles starting the batch. (See VAR1 writeup for statistical formulae.)

Called from: BANKR (-3)

Commons required: DET

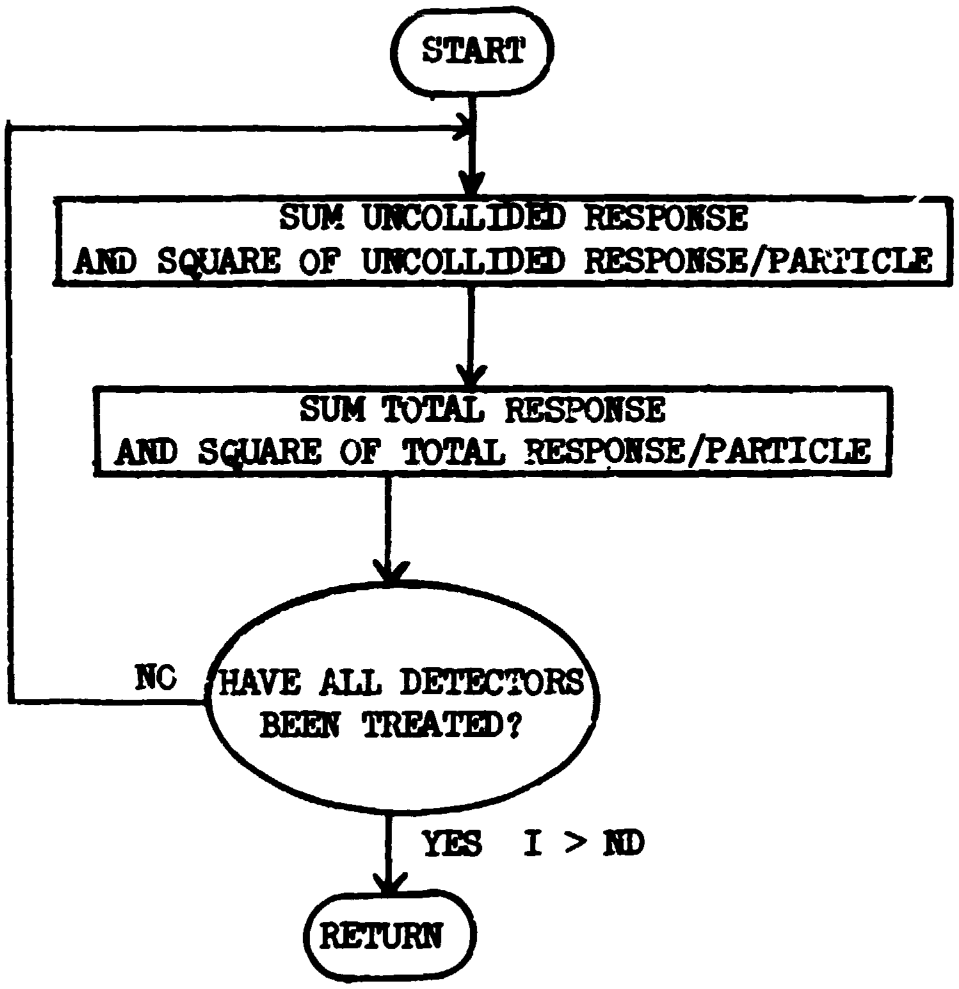
Variables required:

ND, UD(I), SUD(I), SUD2(I), SD(I), SSD(I), SSD2(I) (from common DET, see page 169)

NSØRC - number of particles beginning the batch.

Variables modified: SUD(I), SUD2(I), SSD(I), SSD2(I).

Subroutine NBATCH (NSØRC)


START
SUM UNCOLLIDED RESPONSE
AND SQUARE OF UNCOLLIDED RESPONSE/PARTICLE
SUM TOTAL RESPONSE
AND SQUARE OF TOTAL RESPONSE/PARTICLE
HAVE ALL DETECTORS BEEN TREATED?
NO
YES I > ND
RETURN

```
      SUBROUTINE NBATCH(NSORC)
C
C     NBATCH  SUMS BATCH-QUANTITIES OF INTEREST OVER ALL BATCHES.
C
      COMMON /DET/ ND,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      DO 5 I=1,ND
      F = UD(I)
      SUD(I) = SUD(I) + F
      SUD2(I) = SUD2(I) + F**2/NSORC
      G = SD(I)
      SSD(I) = SSD(I) + G
    5 SSD2(I) = SSD2(I) + G**2/NSORC
      RETURN
      END
```

Subroutine NRUN (NRUNS, NQUIT)

This routine is called at the end of each run (consisting of NRUNS batches of NSTRT particles in this case). The calculated quantities are normalized and output, along with fractional standard deviations.

Called from: BANKR(-4)

Subroutines called:

VAR1 - calculates fractional standard deviations.

Commons required: DET, USER

Variables required:

NRUNS - number of batches completed (note the NITS in common USER is the requested number of batches, not necessarily the actual number completed),

NQUIT - number of runs remaining plus one, or negative of the number of runs completed when an execution time kill occurs,

NSTRT - number of particles per batch,

ND, SUD(I), SUD2(I), SSD(I), SSD2(I), NN(I), RAD(I), NSCAPE (from common DET, see page 169)

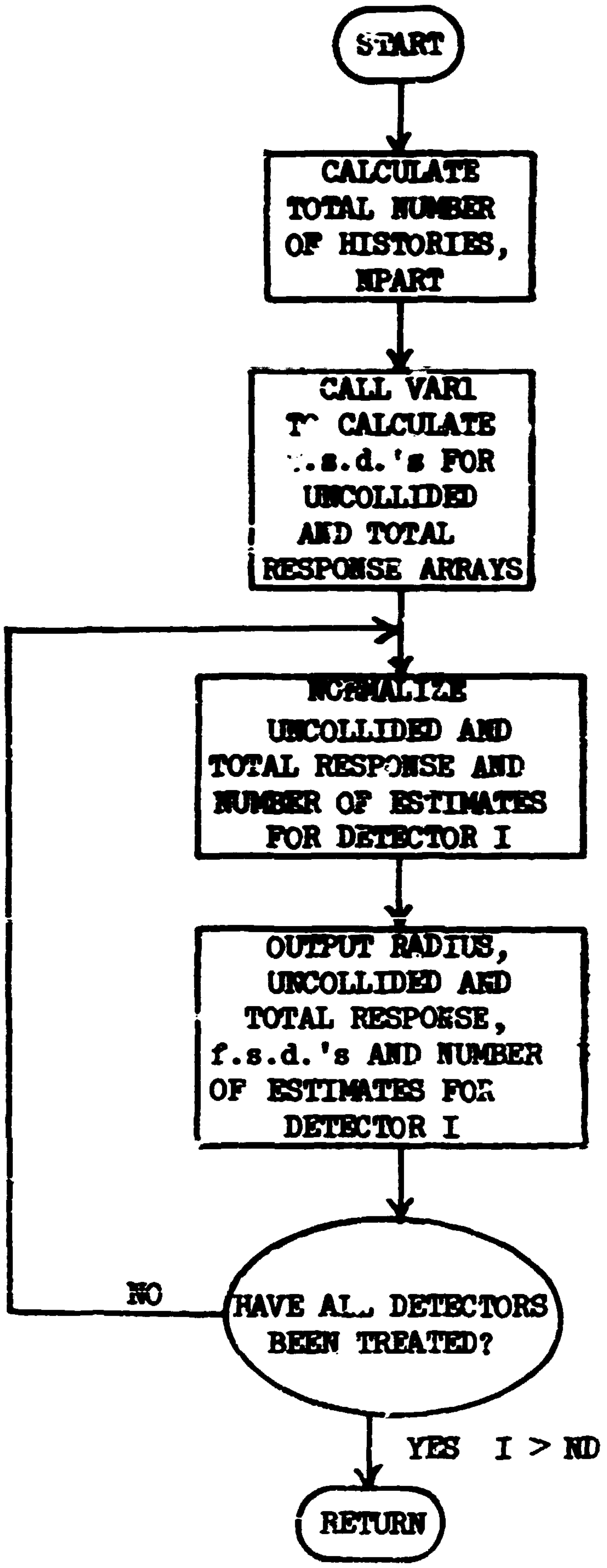
Variables changed:

SUD2(I) }
SSD2(I) } converted to fractional standard deviations by VAR1

SUD(I) }
SSD(I) } normalized to unit source particle

Variables output: RAD(I), SUD(I), SUD2(I), SSD(I), SSD2(I), FIN (NN(I) normalized)

Subroutine NRUN (NRUN,NQUIT)


START
CALCULATE TOTAL NUMBER OF HISTORIES, NPART
CALL VAR1 TO CALCULATE f.s.d.'s FOR UNCOLLIDED AND TOTAL RESPONSE ARRAYS
NORMALIZE UNCOLLIDED AND TOTAL RESPONSE AND NUMBER OF ESTIMATES FOR DETECTOR I
OUTPUT RADIUS, UNCOLLIDED AND TOTAL RESPONSE, f.s.d.'s AND NUMBER OF ESTIMATES FOR DETECTOR I
HAVE ALL DETECTORS BEEN TREATED?
NO
YES I > ND
RETURN

```
      SUBROUTINE NRUN(NRUNS,NQUIT)
C
C     SUBROUTINE  NRUN  SUMS OVER ALL BATCHES, AND CALCULATES AND
C       OUTPUTS QUANITITIES OF INTEREST AFTER A COMPLETE RUN.
C
      COMMON /DET/ ND,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      COMMON /USER/ AGSTRT,WTSTRT,XSTRT,YSTRT,ZSTRT,DFF,EBOTN,EBOTG,
     1 TCUT,IO,I1,IADJM,NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,NLAST,
     2 NLEFT,NMGP,NMTG,NSTRT
      NPART = NRUNS*NSTRT
      FNB = 1.0/NPART
      WRITE (IO,1000)
 1000 FORMAT(1H1,52X,'4 PI R**2 RESPONSE'
     1                                /1H0,6X,'RADIUS',14X,'UNCOLL',16X,
     1'FSD',15X,'TOTAL',17X,'FSD',12X,'FRACTICN OF'/1H ,25X,'RESPONSE',
     213X,'UNCOLL',13X,'RESPONSE',13X,'TOTAL',12X,'CROSSINGS')
      CALL VAR1(SUD(1),SUD2(1),ND,NRUNS,NPART)
      CALL VAR1(SSD(1),SSD2(1),ND,NRUNS,NPART)
      DO 5 I=1,ND
      FIN = NN(I)
      FIN = FIN*FNB
      SUD(I) = SUD(I)*FNB
      SSD(I) = SSD(I)*FNB
 5    WRITE (IO,1010) RAD(I),SUD(I),SUD2(I),SSD(I),SSD2(I),FIN
 1010 FORMAT(1H ,2X,2(E10.4,11X),F 8.5,E21.4,2F19.5)
      RETURN
      END
```

Subroutine SCØRIN

This routine is called by subroutine INPUT for the user to input necessary analysis data. In this sample, a title card, the number and radii of detectors, and values of the response function are read in and output.

Called from: INPUT

Commons required: USER, DET

Variables input and output:

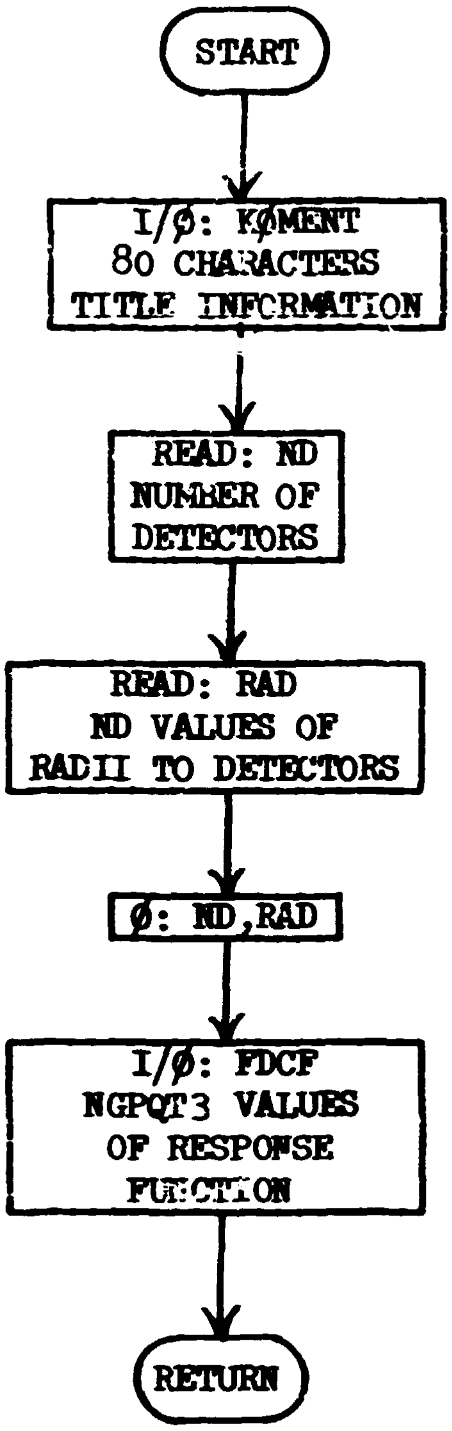
KØMENT - 80 hollerith characters,

ND - number of detectors,

RAD(I) - radii for each of ND detectors,

FDCF(I) - NGPQT3 (=NGPQTN in this case) values of the response function.

Subroutine SCØRIN



```
      SUBROUTINE SCORIN
C
C     ANALYSIS INPUT DATA ARE READ INTO SCORIN
C
      COMMON /USER/ AGSTRT,WTSTRT,XSTRT,YSTRT,ZSTRT,DFF,EBOTN,EBOTG,
     1 TCUT,I2,I1,IADJM,NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,NLAST,
     2 NLEFT,NMGP,NMTG,NSTRT
      COMMON /DET/ ND,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      DIMENSION KOMENT(20)
      READ (I1,1000) KOMENT
 1000 FORMAT (20A4)
C
C     READ IN PROBLEM OUTPUT PARAMETERS
C
C     WHERE
C        ND = NUMBER OF DETECTORS
C
      READ (I1,1010) ND
 1010 FORMAT (8I10)
C
C     READ IN DETECTOR POSITIONS (MUST CORRESPOND TO MEDIA BOUNDARIES)
C     DETECTOR MUST NOT BE PLACED AT THE LAST MEDIUM BOUNDARY.
C
      READ (I1,1020) (RAD(I),I=1,ND)
 1020 FORMAT (7E10.4)
C
C     READ IN NGPQT3 VALUES OF THE RESPONSE FUNCTIONS
C
      READ (I1,1020) (FDCF(I),I=1,NGPQT3)
C
      WRITE (I2,1030) KOMENT
 1030 FORMAT (1H1,20A4)
      WRITE (I2,1040) ND,(RAD(I),I=1,ND)
 1040 FORMAT (1H0,19HNUMBER OF DETECTORS,I4/20H      DETECTOR RADII,
     1 (1P5E14.3))
      WRITE (I2,1050) (FDCF(I),I=1,NGPQT3)
 1050 FORMAT (1H2,'   RESPONSE FUNCTION                ',/,6(1PE14.3))
      RETURN
      END
```

Subroutine SDATA

Called by BANKR(1), from MSØUR, for each source collision, this routine calculates uncollided response for each detector.

Called from: BANKR(1)

Subroutines called: NSIGTA

Functions required: EXP (library)

Commons required: USER, DET, NUTRØN

Variables required:

IG - energy group index,

NMED - medium number,

TSIG - total cross section provided by NSIGTA,

ND - number of detectors,

RAD(I) - array of detector radii,

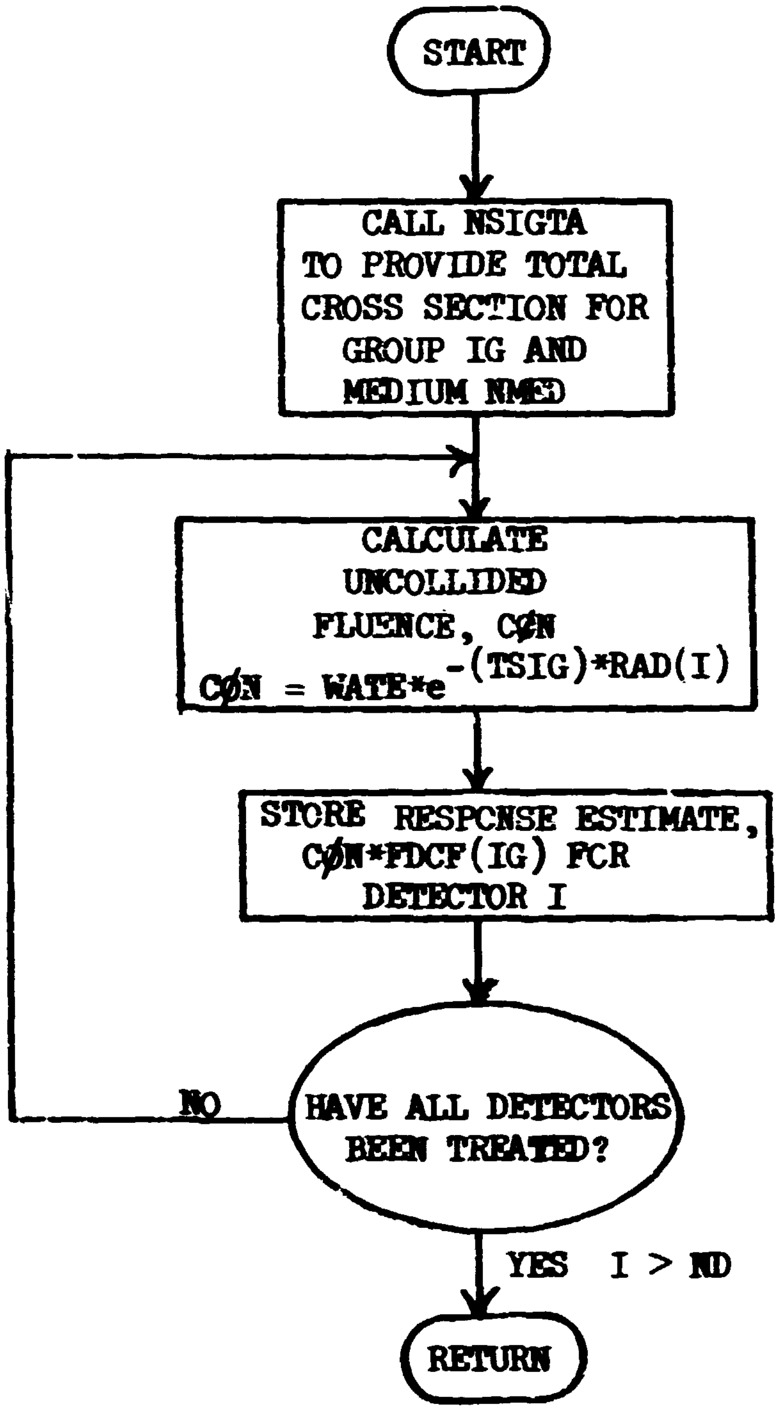
WATE - neutron weight,

FDCF(I) - array of response functions

Variables modified:

UD(I) - array of uncollided responses.

Subroutine SDATA


START
CALL NSIGTA TO PROVIDE TOTAL CROSS SECTION FOR GROUP IG AND MEDIUM NMED
CALCULATE UNCOLLIDED FLUENCE, CØN
CØN = WATE*e^-(TSIG)*RAD(I)
STORE RESPCNSE ESTIMATE, CØN*FDCF(IG) FCR DETECTOR I
HAVE ALL DETECTORS BEEN TREATED?
NO
YES I > ND
RETURN

```
      SUBROUTINE SDATA
C
C     SUBROUTINE  SDATA  CALCULATES UNCOLLIDED QUANTITIES OF INTEREST AT
C       EACH DETECTOR POSITION FOR EACH BATCH.
C
      COMMON /DET/ NO,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      COMMON /USER/ AGSTRT,WTSTRT,XSTRT,YSTRT,ZSTRT,DFF,EBOTN,EBOTG,
     1 TCUT,I0,I1,IADJM,NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,NLAST,
     2 NLEFT,NMGP,NMTG,NSTRT
      COMMON /NUTRON/ NAME,NAMEX,IG,IGO,NMED,MEDOLD,NREG,U,V,W,UOLD,VOLD
     1 ,WOLD,X,Y,Z,XOLD,YOLD,ZOLD,WATE,OLDWT,WTBC,BLZNT,BLZON,AGE,OLDAGE
      CALL NSIGTA(IG,NMED,TSIG,XI)
      DO 5 I=1,NO
      XI=RAD(I)
      CON=WATE*EXP (-TSIG*XI)
      COND = CON*FDCF(IG)
      UD(I) = UD(I) + COND
 5    CONTINUE
      RETURN
      END
```

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 source 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 INPUT,

ITSTR - a switch which indicates whether fission is an original source particle or a daughter (irrelevant in this problem),

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,

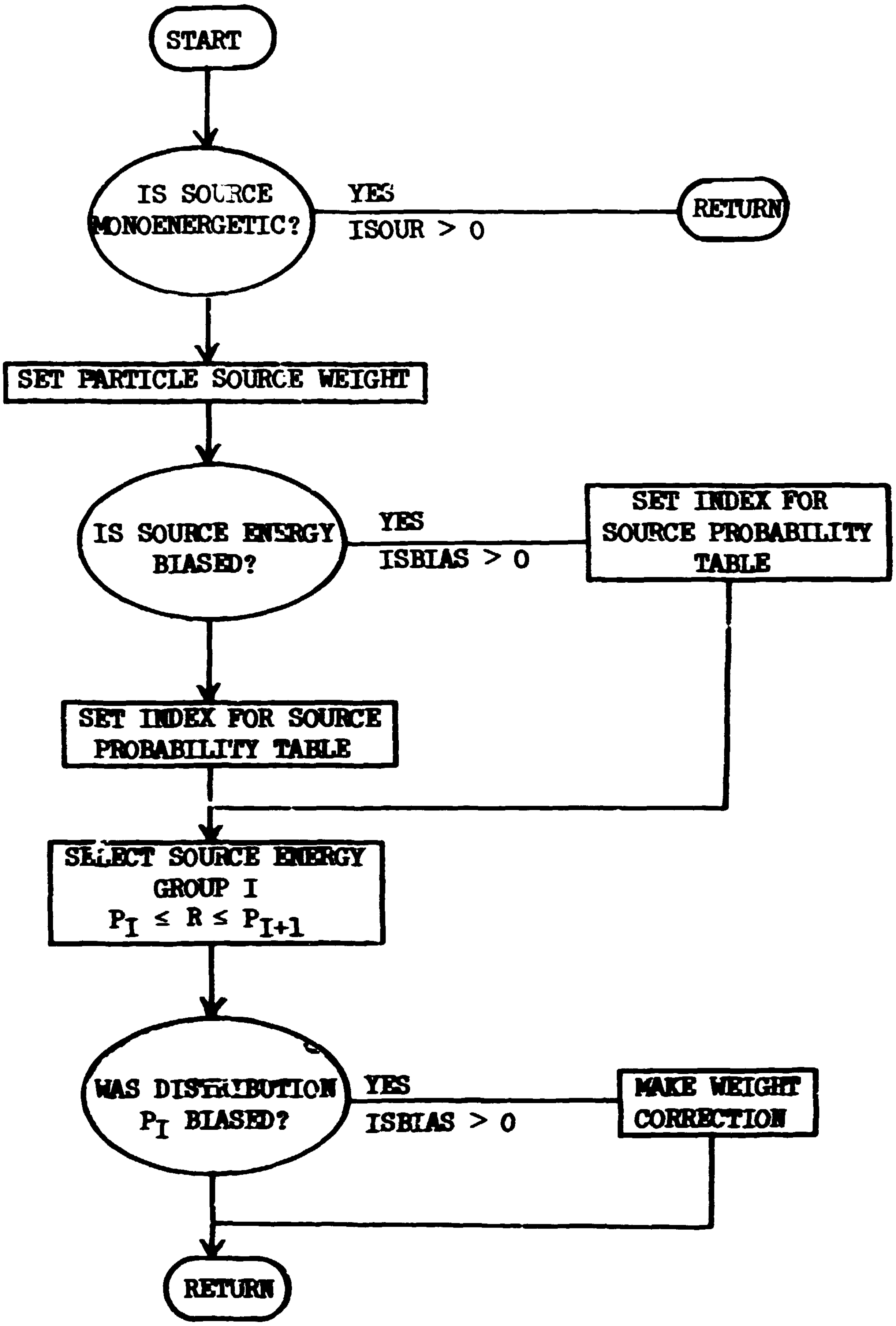
IG - particle energy group.

Significant internal variables:

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

Limitations: This version only selects an energy group.

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


START
IS SOURCE MONOENERGETIC?
YES
ISOUR > 0
RETURN
SET PARTICLE SOURCE WEIGHT
IS SOURCE ENERGY BIASED?
YES
ISBIAS > 0
SET INDEX FOR SOURCE PROBABILITY TABLE
SET INDEX FOR SOURCE PROBABILITY TABLE
SELECT SOURCE ENERGY GROUP I
$P_I \le R \le P_{I+1}$
WAS DISTRIBUTION P_I BIASED?
YES
ISBIAS > 0
MAKE WEIGHT CORRECTION
RETURN

```
      SUBROUTINE SOURCE(IG,U,V,W,X,Y,Z,WATE,NED,AG,ISOUR,ITSTR,NGPQT3,  SOURC 10
     1 DDF,ISBIAS,NMTG)                                                 SOURC 11
C                                                                       SOURC 20
C    IF ITSTR=0, MUST PROVIDE IG,X,Y,Z·U,V,W,WATE AND AG IF DESIRED TO BESOURC 30
C      DIFFERENT FROM CARD VALUES (WHICH ARE THE VALUES INPUT TO SOURCE) SOURC 40
C    IF ITSTR=1, IG IS THE GRP NO. CAUSING FISSION, MUST PROVI(= NEW IG  SOURC 50
C      THIS VERSION OF  SOURCE  SELECTS INITIAL GROUP FROM THE FISSION SPECTRUM.
C                                                                       SOURC 70
      COMMON WTS(1)                                                     SOURC 80
      IF(ISOUR)5,5,50                                                   SOURC 90
 5    WATE=DDF                                                          SOUR 100
      IF (ISBIAS) 10,10,15                                              SOUR 110
 10   NWT = 2*NMTG                                                      SOUR 120
      GO TO 20                                                          SOUR 130
 15   NWT = 3*NMTG                                                      SOUR 140
 20   R = FLTRNF(R)                                                     SOUR 150
      DO 25 I=1,NGPQT3                                                  SOUR 160
      IF (R.LE.WTS(I+NWT)) GO TO 30                                     SOUR 170
 25   CONTINUE                                                          SOUR 180
 30   IG=I                                                              SOUR 190
      IF (ISBIAS) 50,50,35                                              SOUR 200
 35   IF (I-1) 50,40,45                                                 SOUR 210
 40   WATE = WATE*WTS(2*NMTG+1)/WTS(3*NMTG+1)                           SOUR 220
      RETURN                                                            SOUR 230
 45   WATE = WATE*(WTS(2*NMTG+I)-WTS(2*NMTG+I-1))/(WTS(3*NMTG+I)-WTS(3*NSOUR 240
     1MTG+I-1))                                                         SOUR 241
 50   RETURN                                                            SOUR 250
      END                                                               SOUR 260
```

Subroutine STBTCH (NBATCH)

The arrays used to accumulate uncollided and total response are zeroed by this routine. In addition, if NBATCH = 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 from: MØRSE

Subroutines called: ERRØR (library)

Commons required: DET, USER

Variables required:

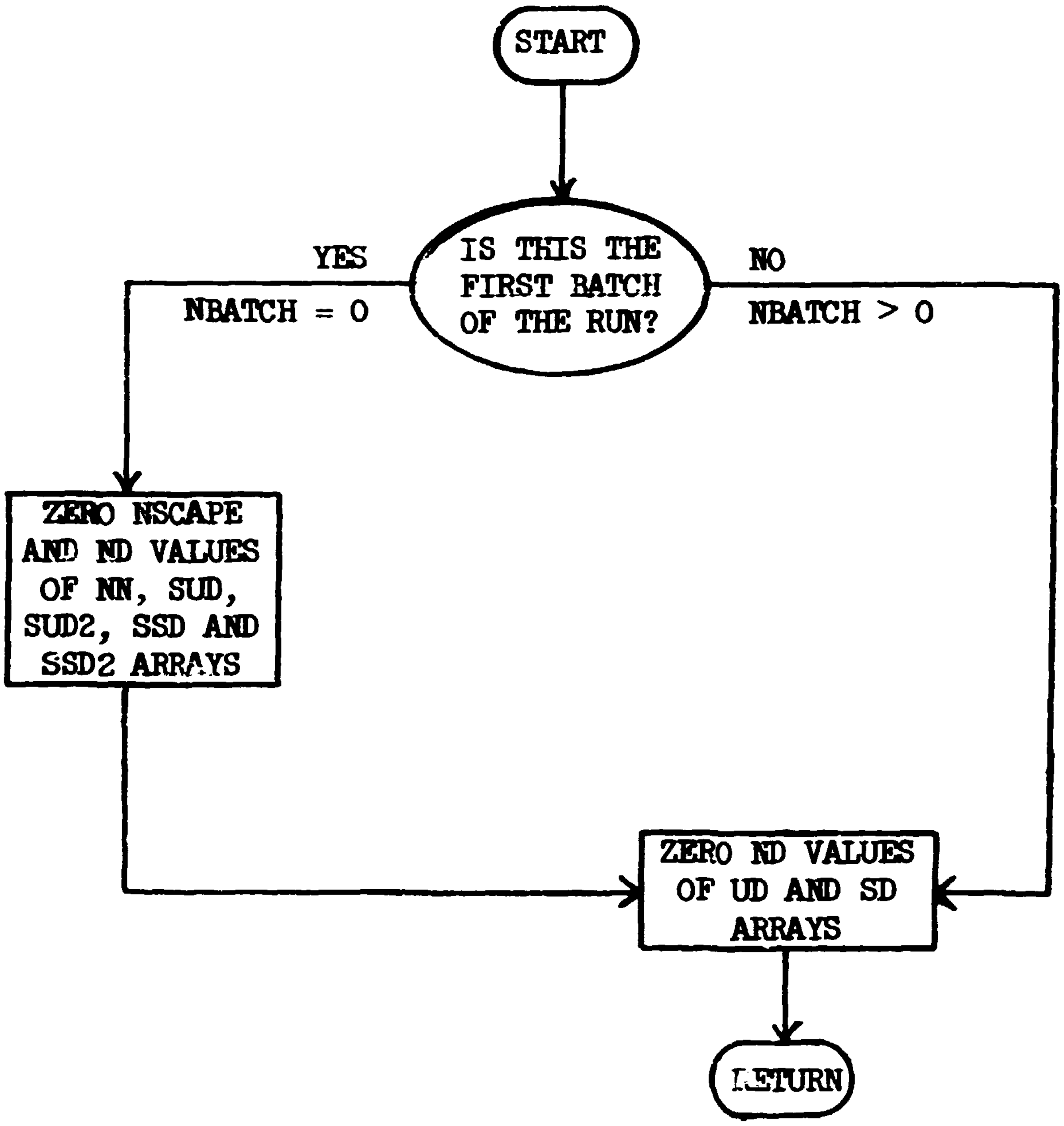
NBATCH - batch number less one,

ND - number of detectors.

Variables modified:

NSCAPE, NN(I), SUD(I), SUD2(I), SSD(I), SSD2(I), SD(I), UD(I) (from common DET, see page 169).

Subroutine STB̄CH (NBATCH)


START
IS THIS THE FIRST BATCH OF THE RUN?
YES
NBATCH = 0
NO
NBATCH > 0
ZERO NSCAPE AND ND VALUES OF NN, SUD, SUD2, SSD AND SSD2 ARRAYS
ZERO ND VALUES OF UD AND SD ARRAYS
RETURN

```
      SUBROUTINE STBTCH(NBATCH)
C
C     THE FOLLOWING QUANITIES ARE INITIALIZED IN STBATCH
C
C       UD(I) = UNCOLLIDED RESPONSE SUMMED OVER A SINGLE BATCH
C       SUD(I) = SUM OF UNCOLLIDED RESPONSE SUMMED OVER ALL BATCHES
C       SUD2(I) = SUM OF UD(I)**2
C       SD(I) = TOTAL RESPONSE SUMMED OVER A SINGLE BATCH
C       SSD(I) = SUM OF TOTAL RESPONSE OVER ALL BATCHES
C       SSD2(I) = SUM OF SD(I)**2
C
C     WHERE
C       I IS THE INDEX FOR DETECTORS (CM)
C
      COMMON /DET/ ND,NSCAPE,RAD(20),NN(20),UD(20),SUD(20),SUD2(20),
     1 SD(20),SSD(20),SSD2(20),FDCF(100)
      COMMON /USER/ AGSTRT,WTSTRT,XSTRT,YSTRT,ZSTRT,DFF,EBOTN,EBOTG,
     1 TCUT,I0,I1,IADJM.NGPQT1,NGPQT2,NGPQT3,NGPQTG,NGPQTN,NITS,NLAST,
     2 NLEFT,NMGP,NMTG,NSTRT
      IF (NBATCH) 5,10,20
    5 CALL ERROR
   10 NSCAPE = 0
      DO 15 I=1,ND
      NN(I)=0
      SUD(I) = 0.0
      SUD2(I) = 0.0
      SSD(I) = 0.0
   15 SSD2(I) = 0.0
   20 DO 25 I=1,ND
      SD(I) = 0.0
   25 UD(I) = 0.0
      RETURN
      END
```

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. In this sample, it is used to print out the first 50 random numbers for assistance to users trying to duplicate the random number generator. Note that the starting random number is saved and restored before returning.

Called from: BA"KR(-1)

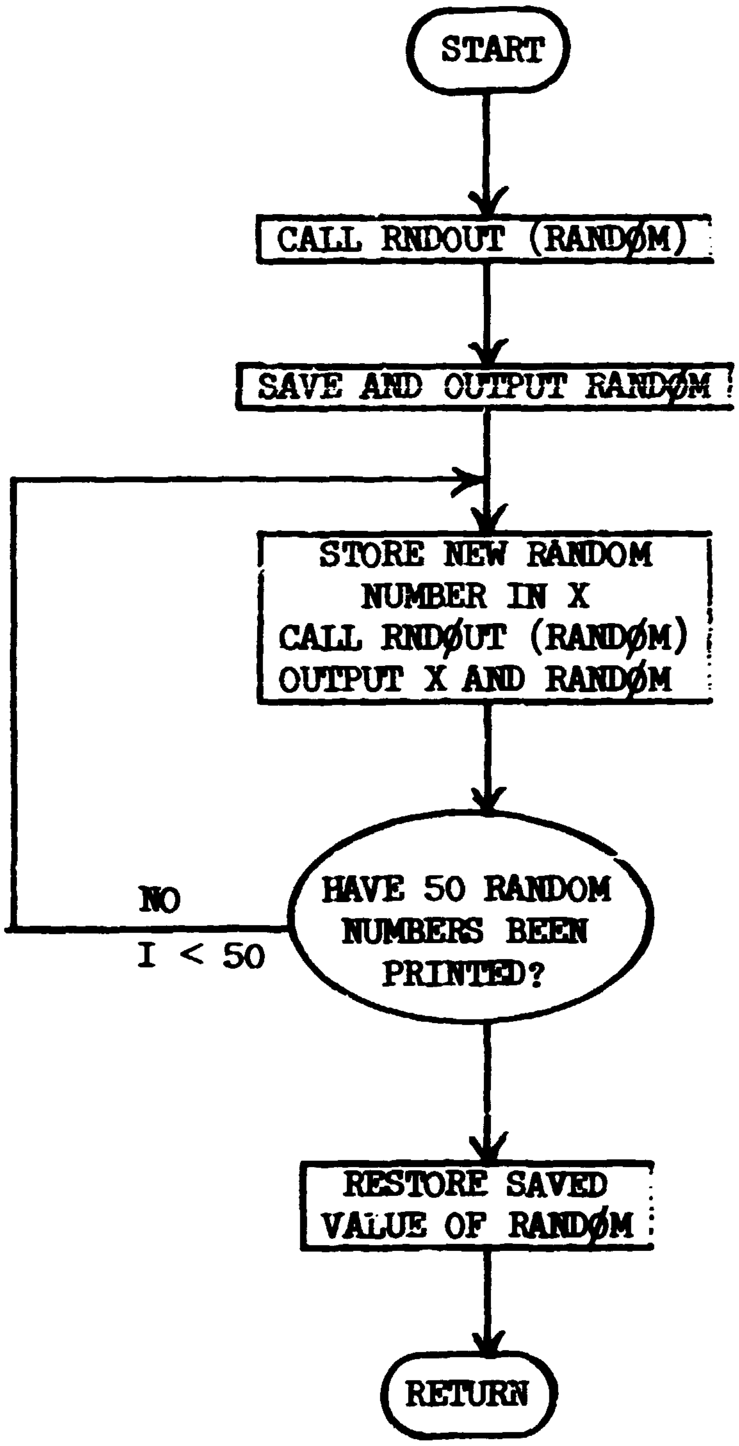
Subroutines called:

RNØOUT

RNDIN

Functions used: FLTRNF

Subroutine STRUN


START
CALL RNDOUT (RANDØM)
SAVE AND OUTPUT RANDØM
STORE NEW RANDOM NUMBER IN X CALL RNDØUT (RANDØM) OUTPUT X AND RANDØM
HAVE 50 RANDOM NUMBERS BEEN PRINTED?
NO
I < 50
RESTORE SAVED VALUE OF RANDØM
RETURN

```
      SUBROUTINE STRUN                                                  STRUN 10
C * * THIS ROUTINE IS ENTERED ONLY AT THE BEGINNING OF EACH SET OF NITS BATCHES
C * * * THIS VERSION PRINTS A LIST OF THE FIRST FEW RANDOM NUMBERS SO THAT USERS
C * * * OF VARIOUS MACHINES MAY DUPLICATE THE RANDOM NUMBER SEQUENCE    STRUN 40
      REAL*8 RANDOM,RSAVE                                               STRUN 50
      CALL RNDOUT(RANDOM)                                               STRUN 60
      RSAVE = RANDOM                                                    STRUN 70
      WRITE (6,1000) RANDOM                                             STRUN 80
 1000 FORMAT (' THE INITIAL RANDOM NUMBER, IN HEX, IS ',Z16,/           STRUN 90
     1  ' THE NEXT 50 NUMBERS FOLLOW'/)                                 STRUN 91
      DO 5 I=1,50                                                       STRU 100
      X = FLTRNF(X)                                                     STRU 110
      CALL RNDOUT(RANDOM)                                               STRU 120
 5    WRITE (6,1005) X,RANDOM                                           STRU 130
 1005 FORMAT (F20.8,4X,Z12)                                             STRU 140
      RANDOM = RSAVE                                                    STRU 150
      CALL RNDIN(RANDOM)                                                STRU 160
      RETURN                                                            STRU 170
      END                                                               STRU 180
```

Subroutine VAR1 (SX, SX2, M, NBAT, NPART)

This routine calculates variances and fractional standard deviations (f.s.d.) for batch statistics allowing for unequally weighted batches. The formula for the 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 the ith batch,

x_i = accumulated estimate in the ith 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 jth history in the ith 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

$$\text{f.s.d.} = \sqrt{\sigma_{\bar{x}}^2}/\bar{x} .$$

Note that the routine must be called before the array SX(=n$\bar{x}$) is normalized.

Called from: NRUN

Functions required: SQRT, ABS (library functions).

Variables required:

SX(I) - array of values of $n\bar{x} = \sum_{i=1}^{N} n_i x_i$, SX2(I) - array of values of $\sum_{i=1}^{N} n_i x_i^2$,

M - number of elements in SX and SX2,

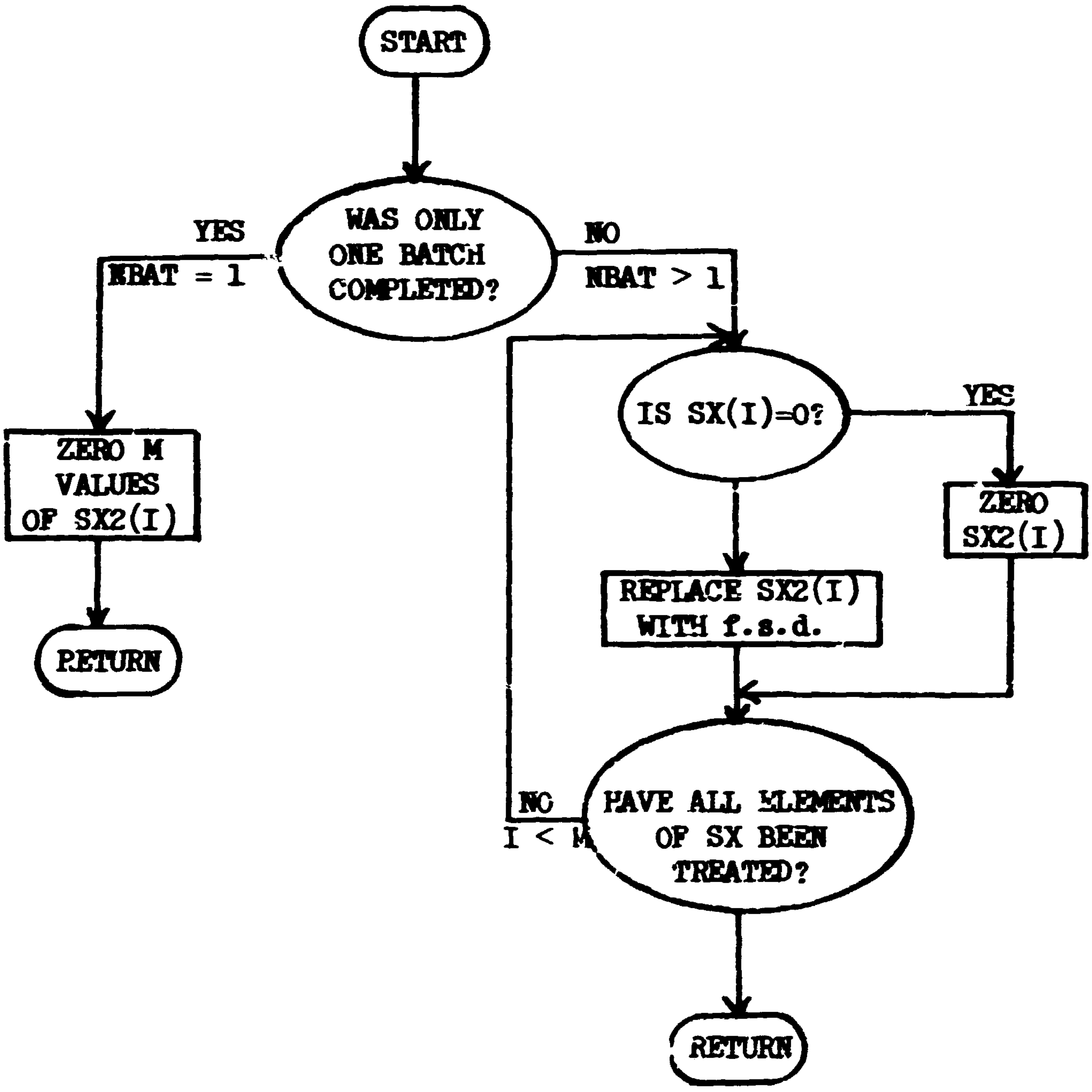
NBAT = N,

NPART = n.

Variables changed:

SX2(I) changed to f.s.d.

Subroutine VAR1 (SX,SX2,M,NBAT,NPART)



```
      SUBROUTINE VAR1(SX,SX2,M,NBAT,NPART)
C     NBAT IS THE NO. OF INDEPENDENT BATCHES
C     NPART IS THE TOTAL NUMBER OF PARTICLES PROCESSED
C     IT IS ASSUMEC THAT THE SUMSQ ARRAY HAS ACCUMULATED THE NUMBER OF PARTICLES
C       TIMES THE SQUARE OF THE BATCH AVERAGE (THIS IS OBTAINED BY DIVIDING
C       THE SQUARED BATCH SUM BY THE NUMBER OF PARTICLES STARTING THE BATCH)
      DIMENSION SX(M),SX2(M)
      IF (NBAT-1) 5,5,15
    5 DO 10 I=1,M
   10 SX2(I) = 0.0
      RETURN
   15 DO 30 I=1,M
      IF (SX(I)) 25,20,25
   20 SX2(I) = 0.0
      GO TO 30
   25 SX2(I) = (SX2(I)/NPART - (SX(I)/NPART)**2)/(NBAT-1.)
      SX2(I) = SQRT(ABS(SX2(I)))/SX(I)*NPART
   3C CONTINUE
      RETURN
      END
```

Sample Problem

The fast-neutron fluence at several radial distances is calculated 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 special spherical geometry 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 X. Splitting, Russian roulette, and path length stretching were also implemented.

The problem input and output are listed as follows:

Table X. Fission Spectrum in 14-Group Structure

Group No.	Energy Limits (MeV)	Fraction of Source Neutrons
1	15.0-12.21	1.5529(-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	0.1111-0.3308	1.5489(-2)

[a] Read as 1.5529 x 10^{-4}.

CODED BY ______________ DATE ______ REQUEST NO. ______

FORTRAN

PROBLEM ______
PROGRAM ______
PAGE ____ OF ____

```
MARSE SAMPLE PROBLEM POINT FISSION SOURCE IN AIR                        CARD A
  200  400   10    1   12    0   22   22    0    0    5    1    0        CARD B
    0   14    0    0 1.0       0.0       0.0       1.0       2.2E    +5 CARD C
      0.0       0.0       0.0       0.0       0.0       0.0       0.0   CARD D
1.5579  -4 8.9335  -4 3.4756  -3 1.3905  -2 3.4557  -2 8.5047  -2 1.0724  -1CARDS E
9.5968  -2 2.3156  -2 1.2030  -1 2.1502  -1 1.9837  -1 1.4036  -1 1.5459  -2CARDS E
1.5000  +7 1.2214  +7 1.0000  +7 8.1275  +6 6.2600  +6 4.9258  +6 4.0667  +6CARDS F
3.0119  +6 2.4659  +6 2.3500  +6 1.8268  +6 1.1088  +6 5.5020  +5 1.1109  +5CARDS F
8.3596  +3 5.5274  +2 1.6130  +2 2.9020  +1 1.0677  +1 8.0590  +1 1.1265  +1CARDS F
0.414                                                                   CARDS F
000035FA931A                                                            CARD H
    1    1    1    0    0    1   13                                     CARD I
    0    0    0    0    0    0 1.0    +1 1.0    -2 1.0    +0 5.0    -1   CARD J
   -1                                                                   -1 CARD
    0    0    0    0                                                    CARD L
    1 3.0    +3
    2 5.0    +3
    1 7.5    +3
    2 1.0    +4
    1 1.5    +4
```

CODED BY ______ PROBLEM ______

DATE ______ REQUEST NO. ______ PROGRAM ______

FORTRAN

PAGE ______ OF ______

```
    2 2.0    +4
    1 3.0    +4
    2 6.0    +4
    1 9.0    +4
    2 1.2    +5
    1 1.5    +5
    2 1.0    +6
         ( BLANK CARD HERE)
1.0     +6
         ( BLANK CARD HERE)
22 GROUP AIR CROSS SECTIONS --- PS --- DENSITY = 1.29 G/L
   22   22    0    0   22   22    1    1    1    6    3    0
    0    0    0    0    0    0    0
         ( 22 GROUP AIR CROSS SECTIONS IN ANISN FORMAT HERE)
    1   -1 1.16
ANALYSIS INPUT DATA                                                     CARD  AA
         6                                                              CARD  BB
    1.0 +4    2.0 +4    3.0 +4    6.0 +4    9.0 +4   12.0 +4            CARD  CC
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                   CARDS DD
```

```
MORSE SAMPLE PROBLEM  POINT FISSION SOURCE IN AIR
THIS CASE WAS BEGUN ON TUESDAY, AUGUST 4, 1970

NSTRT=  200      NMOST=  400      NITS=  10        NQUIT=    1       NGPQTN=  13
NGPQTG=   0      NMGP=   22       NMTG=   22       NCOLTP=   0       IADJM=    0
      MAXIMUM EXECUTION TIME =    5 MINUTES        MEDIA=    1       MEDALB=    0

ISOUR=     0   NGPFS=   14   ISBIAS=    0   NRESP=    0
  WTSTRT=0.1000E 01   EBOTN=0.0          EBOTG=0.0          TCUT=0.1000E 01   VELTH=0.2200E 06

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

DDF IS DIFFERENT FROM WTSTRT, DDF = 0.98451E 00

SPECTRUM OF CUMULATIVE GROUP PROBABILITIES

    FS( 1)=0.1582E-03           FS( 2)=0.1066E-02
    FS( 3)=0.4599E-02           FS( 4)=0.1872E-01
    FS( 5)=0.5382E-01           FS( 6)=0.8942E-01
    FS( 7)=0.1984E 00           FS( 8)=0.2887E 0C
    FS( 9)=0.3123E 00           FS(10)=0.4345E 00
    FS(11)=0.6559E 00           FS(12)=0.8574E 00
    FS(13)=0.1000E 01           FS(14)=0.0

GROUP PARAMETERS, GROUP NUMBERS GREATER THAN   22 CORRESPOND TO SECONDARY PARTICLES
```

```
      GROUP       UPPER EDGE           VELOCITY
                     (EV)              (CM/SEC)
        1        0.1500E 08         0.5102E 10
        2        0.1221E 08         0.4609E 10
        3        0.1000E 08         0.4171E 10
        4        0.8187E 07         0.3730E 10
        5        0.6360E 07         0.3291E 10
        6        0.4966E 07         0.2939E 10
        7        0.4066E 07         0.2602E 10
        8        0.3012E 07         0.2289E 10
        9        0.2466E 07         0.2146E 10
       10        0.2350E 07         0.1999E 10
       11        0.1827E 07         0.1675E 10
       12        0.1108E 07         0.1259E 10
       13        0.5502E 06         0.7952E 09

    INITIAL RANDOM NUMBER = 000035FA731A

    NSPLT= 1      NKILL= 1      NPAST= 1      NOLEAK= 0      IEBIAS= 0      MXREG=  1      MAXGP= 13

    WEIGHT STANDARDS FOR SPLITTING AND RUSSIAN ROULETTE AND PATHLENGTH STRETCHING PARAMETERS

     NGP1   NDG  NGP2   NRG1  NDRG  NRG2     WTHIH1       WTLOW1       WTAVE1         XNU
       0     0     0      0     0     0    0.1000E 02   0.1000E-01   0.1000E 00   0.5000E 00

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

```
SPHERICAL GEOM

 MEDIUM     RADIUS
    1     0.30000D 04
    2     0.50000D 04
    1     0.75000D 04
    2     0.10000D 05
    1     0.15000D 05
    2     0.20000D 05
    1     0.30000D 05
    2     0.60000D 05
    1     0.70000D 05
    2     0.90000D 05
    1     0.12000D 06
    2     0.15000D 06
    1     0.10000D 07

 REGION     RADIUS
    1     0.10000D 07

NGEOM=    529,    NGLAST=    528
```

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

NUMBER OF NEUTRON GROUPS          22

NUMBER OF NEUTRON DOWNSCATTERS    22

NUMBER OF GAMMA GROUPS             0

NUMBER OF GAMMA DOWNSCATTERS       0

NUMBER OF INPUT GROUPS            22

NUMBER OF INPUT DOWNSCATTERS      22

NUMBER OF MEDIA                    1

NUMBER OF INPUT ELEMENTS           1

NUMBER OF MIXING ENTRIES           1

NUMBER OF COEFFICIENTS             6

NUMBER OF ANGLES                   3

ADJOINT SWITCH                     0


OPTIONS ARE ACCEPTED IF VARIABLE IS GT 0
IRDSG=    0      ISTR=    0      IFMU=    0    IMOM=    0    IPRIN=    0      IPUN=    0
 IDTF=    0     ISTAT=    0    IXTAPE=    0
CROSS SECTIONS START AT       529
LAST LOCATION USED (PERM)    2435
LAST LOCATION USED (TEMP)    4228

ELEMENT    1   APPEARS IN MEDIA    1  WITH DENSITY   1.1600E 00
```

```
                        CROSS SECTIONS FOR MEDIA      1
GROUP      SIGT        SIGS      PNABS    GAMGEN  NU*FISS  DOWNSCATTER PROBABILITY
   1  8.149E-05  6.753E-05  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  0.8271  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     0.0
   3  6.718E-05  5.516E-05  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  0.8390  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  0.8562  0.0       0.0      0.5150  0.4087  0.0528  0.0042  0.0012  0.0053  0.0065  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  0.7994  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  0.8463  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
   8  7.212E-05  6.391E-05  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
   9  6.210E-05  5.819E-05  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  0.9510  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  0.9736  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     0.0
  12  1.175E-04  1.159E-04  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     0.0
  13  1.850E-04  1.847E-04  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  0.9998  0.0       0.0      0.9614  0.0386  0.0     0.0     0.0     0.0     0.0     0.0
                                                        0.0
  15  4.159E-04  4.156E-04  0.9992  0.0       0.0      0.9235  0.0765  0.0     0.0     0.0     0.0     0.0     0.0

  16  4.432E-04  4.424E-04  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  0.9962  0.0       0.0      0.8941  0.1059  0.0     0.0     0.0     0.0
  18  4.667E-04  4.637E-04  0.9935  0.0       0.0      0.8682  0.1318  0.0     0.0     0.0
  19  4.745E-04  4.691E-04  0.9887  0.0       0.0      0.8974  0.1026  0.0     0.0
  20  4.826E-04  4.732E-04  0.9805  0.0       0.0      0.8734  0.1266  0.0
  21  4.884E-04  4.730E-04  0.9684  0.0       0.0      0.8779  0.1221
  22  5.407E-04  4.735E-04  0.8757  0.0       0.0      1.0000

BANKS START AT       2436
LAST LOCATION USED   7235
```

```
ANALYSIS INPUT DATA

NUMBER OF DETECTORS    7
     DETECTOR RADII     1.000E 04     2.000E 04     3.000E 04     6.000E 04     7.000E 04
    9.000E 04     1.200E 05




  RESPONSE FUNCTION
   1.000E 00     1.000E 00     1.000E 00     1.000E 00     1.000E 00     1.000E 00
   1.000E 00     1.000E 00     1.000E 00     1.000E 00     1.000E 00     1.000E 00
   1.000E 00
```

```
TIME REQUIRED FOR INPUT WAS 6 SECONDS.
YOU ARE USING THE DEFAULT VERSION OF STRUN WHICH DOES NOTHING.
YOU ARE USING THE DEFAULT VERSION OF SOURCE WHICH SETS WATE TO CCF AND PROVIDES AN ENERGY IG.

***START BATCH   1                          RANDOM=C4FFE7112412

SOURCE DATA
    WT=   1.9690E 02   UAVE=  -1.7613E-02   VAVE=  -8.2375E-03   WAVE=   4.5008E-02   AGEAVE=   0.0
    IAVE=   10.38   XAVE=   0.0              YAVE=   0.0              ZAVE=   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       180        0       24        1        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH   2                          RANDOM=A0982979E55A

SOURCE DATA
    WT=   1.9690E 02   UAVE=  -4.1214E-02   VAVE=  -6.1608E-02   WAVE=   1.3603E-02   AGEAVE=   0.0
    IAVE=   10.12   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3749         0      2100         0       175        0       25        1        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 6 SECONDS.

***START BATCH   3                          RANDOM=C8EDA896770A

SOURCE DATA
    WT=   1.9690E 02   UAVE=  -2.8524E-02   VAVE=  -4.2949E-02   WAVE=  -5.5498E-03   AGEAVE=   0.0
    IAVE=   10.08   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3905         0      2179         0       179        0       27        4        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH   4                          RANDOM=28889AE580E2

SOURCE DATA
    WT=   1.9690E 02   UAVE=   1.3272E-02   VAVE=   9.0401E-02   WAVE=  -3.5147E-03   AGEAVE=   0.0
    IAVE=   10.12   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3834         0      2161         0       181        0       22        1        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH   5                          RANDOM=B4092CC5611A

SOURCE DATA
    WT=   1.9690E 02   UAVE=   2.6382E-02   VAVE=   1.5445E-02   WAVE=  -1.5468E-02   AGEAVE=   0.0
    IAVE=   10.64   XAVE=   0.0              YAVE=   0.0              ZAVE=   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 6 SECONDS.

***START BATCH   6                          RANDOM=A2E1102437E2

SOURCE DATA
    WT=   1.9690E 02   UAVE=   1.2133E-02   VAVE=  -2.4994E-02   WAVE=  -2.3430E-02   AGEAVE=   0.0
    IAVE=   10.09   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3968         0      2237         0       189        0       13         2        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH   7                          RANDOM=423CADCCECCA

SOURCE DATA
    WT=   1.9690E 02   UAVE=   5.6579E-02   VAVE=   5.C583E-02   WAVE=   1.6982E-02   AGEAVE=   0.0
    IAVE=    9.99   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     4072         0      2241         0       178        0       24         2        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH   8                          RANDOM=B193B628586A

SOURCE DATA
    WT=   1.9690E 02   UAVE=   6.8224E-03   VAVE=  -6.4666E-02   WAVE=  -3.1843E-02   AGEAVE=   0.0
    IAVE=   10.09   XAVE=   0.0              YAVE=   0.0              ZAVE=   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        1         0        0     3678         0      2111         0       174        0       27         0        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 6 SECONDS.

***START BATCH   9                          RANDOM=91EB73530EAA

SOURCE DATA
    WT=   1.9690E 02   UAVE=  -9.4371E-03   VAVE=  -2.3959E-02   WAVE=  -2.0416E-02   AGEAVE=   0.0
    IAVE=    9.93   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3799         0      2175         0       173        0       27         3        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 7 SECONDS.

***START BATCH  10                          RANDOM=1D6B29364452

SOURCE DATA
    WT=   1.9690E 02   UAVE=   1.5821E-02   VAVE=   6.7391E-02   WAVE=   1.3778E-02   AGEAVE=   0.0
    IAVE=   10.28   XAVE=   0.0              YAVE=   0.0              ZAVE=   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     3683         0      2199         0       174        0       26         0        0

TIME REQUIRED FOR THE PRECEDING BATCH WAS 6 SECONDS.
```

```
                                   4 PI R**2 RESPONSE

        RADIUS        UNCOLL          FSD          TOTAL          FSD        FRACTION OF
                     RESPONSE        UNCOLL       RESPONSE        TOTAL      CROSSINGS
      0.1000E 05    0.3351E 00      0.00704     0.1788E 01      0.03067       1.23650
      0.2000E 05    0.1245E 00      0.01255     0.2029E 01      0.04221       1.30250
      0.3000E 05    0.4924E-01      0.01725     0.1606E 01      0.05644       1.21050
      0.6000E 05    0.3925E-02      0.02827     0.6230E 00      0.07218       0.87250
      0.7000E 05    0.1792E-02      0.03110     0.3905E 00      0.10718       0.73600
      0.9000E 05    0.3960E-03      0.03574     0.1329E 00      0.11209       0.47550
      0.1200E 06    0.4536E-04      0.04082     0.4508E-01      0.21420       0.22200


   TIME REQUIRED FOR THE PRECEDING  10 BATCHES WAS 1 MINUTE, 16 SECONDS.



   NEUTRON DEATHS                          NUMBER                WEIGHT

   KILLED BY RUSSIAN ROULETTE                226           0.15116E 01
   ESCAPED                                     0           0.0
   REACHED ENERGY CUTOFF                    1794           0.13430E 04
   REACHED TIME CUTOFF                         0           0.0


   NUMBER OF SCATTERINGS

   MEDIUM               NUMBER
      1               38010
```

```
REAL  SCATTERING  COUNTERS

ENERGY          REGION   1
 GROUP        NUMBER  WEIGHT
    1              0   C.0
    2              7   2.57E 00
    3             11   7.08E 00
    4             50   5.15E 01
    5            151   1.17E 02
    6            283   2.C7E 02
    7            835   5.69E 02
    8            766   5.32E 02
    9            257   1.68E 02
   10           1617   1.06E 03
   11           4619   3.29E 03
   12           8313   6.12E 03
   13          21101   1.62E 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        REGION  1
 GROUP      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    1.39E 01
  13          18    1.09F 02

NUMBER OF SPLITTINGS PREVENTED BY LACK OF ROOM

ENERGY        REGION  1
 GROUP      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         REGION  1
 GROUP       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            5    3.17E-02
   8            7    3.41E-02
   9            1    4.59E-03
  10            6    4.03E-02
  11           34    2.10E-01
  12           43    2.82E-01
  13          129    9.04E-01

NUMBER OF RUSSIAN ROULETTE SURVIVALS

ENERGY         REGION  1
 GROUP       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    9.98E-03
   8            0    0.0
   9            0    0.0
  10            0    0.0
  11            1    9.94E-03
  12            3    2.24E-02
  13           10    8.41E-02

TOTAL CPU TIME FOR THIS PROBLEM WAS   1.28 MINUTES.

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

VI. Geometry Module

MORSE uses the geometry packages that are used with 05R with minor changes. That is, there are spherical, slab, cylindrical, and general three-dimensional geometry packages that can be used. There are several descriptions of the various geometry routines in the 05R manual[6] and in the helpful hints for 05R user's manual.[7]

Changes were made to all of the GEOM packages to allow for albedo scattering from any material surface and for variable input-output logical units. The GEOM packages are available in double precision for the IBM-360.

The geometry packages may be replaced with any special-purpose geometry routines the user might write. The three main functions of the geometry package are performed by the three subroutines discussed below.

Subroutine JØMIN (NADD, INTAPE, IØTAPE)

This subroutine reads geometry input and NADD is the first location in blank common that may be used for input storage. In the special geometry packages blank common is not used, so NADD is not incremented; otherwise, NADD must be incremented by the storage required by geometry data.

Subroutine LØØKZ (X, Y, Z)

This subroutine determines the block and zone number, medium, and region for the point X, Y, Z. This routine is called from MSØUR to determine the starting region and medium for source particles.

Subroutine GEØM

This is the main executive routine in that it determines the end point of a flight given the starting point, direction cosines or a tentative end point, and the number of mean free paths (or physical distance in any desired units) the particle will travel. It is called from GØMST and the information is transferred through common GEØMC (see Table XI). In the more complicated geometry packages there are many routines that assist subroutine GEØM in determining the collision point.

To facilitate the use of the various geometry packages, a brief description of each is included here, and the input instructions for each are given in Appendix D.

Table XI. Definitions of Variables in Common GEØMC As Found in Subroutine GEØM

Variable	Definition
X2,Y2,Z2	Coordinates at tentative end-of-flight or if the trajectory is in an internal void; X2, Y2, Z2 are the direction cosines of the trajectory.
X1,Y1,Z1	Starting coordinates for the particle.
ETA	Number of mean free paths to be traversed if flight goes to X2, Y2, Z2.
ETAUSD	Number of mean free paths actually traversed after the call to GEØM.
IBLZ	An index to the medium number for the special geometry packages. For GENERAL GEØM, IBLZ is a packed word giving the block and zone of the end of flight.
IBZN	A dummy variable.
MARK	A flag set by GEØM indicating the results of the trajectory calculation. = 1 for completed flight. = 0 for boundary crossing. = -1 for escape. = -2 for entering an internal void.
NMED	Medium number at end of flight or of medium about to be entered at a boundary crossing.
NREG	Region number at end of flight; not set at boundary crossings.

SPHERICAL GEØM*

Spherical GEØM is used to describe up to 20 concentric spheres centered at X = Y = Z = 0. Internal voids may be used in any location and media numbers need not be ordered with increasing radii; however, regions must be numbered consecutively from the center. The medium and regions are bounded by the outer radius input. For example, the first region is interior to the surface of radius R_1. External (pure absorber) voids are not allowed except outside the maximum radius.

Subroutine GØMFLP sets the medium index IBLZ and region number NREG to the values appropriate to the medium re-entered after a reflection. Subroutine NØRML calculates the direction cosines of the normal to the spherical surface.

Subroutines required: GEØM, JØMIN, LØØKZ, GØMFLP, NØRML.
Input instructions are given on page D-1.

*Taken from references 6 and 7.

SLAB GEØM*

SLAB GEØM can be used whenever there are rectangular parallelepipeds with normals to medium and region boundaries parallel to the Z axis. A finite width and height are allowed. A maximum of 20 medium and region boundaries may be used with internal voids (medium 1000) allowed, but external voids (medium 0) are not permitted inside the system. Media may be numbered in any order but regions must be numbered consecutively with the region of lowest Z being region 1. The media and regions may have different internal boundaries but the external boundary must be the same. Subroutines GØMFLP and NØRML provide the same functions for SLAB GEØM as for SPHERICAL GEØM.

Subroutines required: GEØM, JØMIN, LØØKZ, GØMFLP, NØRML.

Input instructions are given on page D-2.

*Originally written by N. A. Betz.

CYLINDRICAL GEØM*

CYLINDRICAL GEØM may be used to describe a series of concentric cylindrical cylindrical surfaces with up to 20 heights and 20 radial boundaries. The radial boundaries may be different for each height interval and internal voids (medium 1000) are allowed. Negative heights, i.e., Z < 0, are not allowed. Media and regions numbers may be used in any order.

Subroutines required: GEØM, JØMIN, LØØKZ, GØMFLP, NØRML, JØM4, JØM5, JØM6, JØM9, JØM10.

Input instructions are given on page D-3.

[*] Originally written by K. D. Franz and W. Morrison.

GENERAL GEØM*

The general three-dimensional geometry package has been described in detail elsewhere. The only limitation of geometry detail that may be treated is that surfaces must be describable by quadratic surfaces.

The description of the system must include a rectangular parallelepiped whose faces are parallel to the XY, YZ, and XZ coordinate planes. This parallelepiped is then divided into zones with planes that extend across the entire system. The zones are divided into blocks with planes parallel to coordinate axes but which extend only across the individual zones. Each block is then divided into sectors by quadratic surfaces with the sector defined by whether the volume is positive or negative with respect to the quadratic surfaces. Each sector may contain only one medium; therefore, if a medium cannot be described by a single quadratic surface, it must be divided into several sectors.

Besides material boundaries, internal (medium 1000) and external (medium 0) voids may be used. If an external void is interior to the system it behaves as a perfect absorber since the particle is assumed to have escaped upon entering.

Region geometry may also be described for use in importance sampling. The block and zone boundaries for region geometry must be identical with the material boundaries. A description of geom input is given on page D-4. A code, PICTURE[13], has been written to aid in debugging both material and region geometry input.

Subroutines required: GEØM, JØMIN, LØØKZ, GØMFLP, NØRML, JØM4, JØM5, JØM6, JØM7, JØM9, JØM10, JØM11, JØM12, JØM13, JØM14, JØM15.

Functions required: JØM8, JØM16, JØM17, LØC.

Figure 4 shows the hierarchy of subroutines for GENERAL GEØM. Detailed descriptions of the various routines are given in references 6 and 7; however, some changes have been made since those reports were written. The two main changes incorporated in Subroutines GØMFLP and NØRML are discussed.

[*] Discussion taken from references 6 and 7.

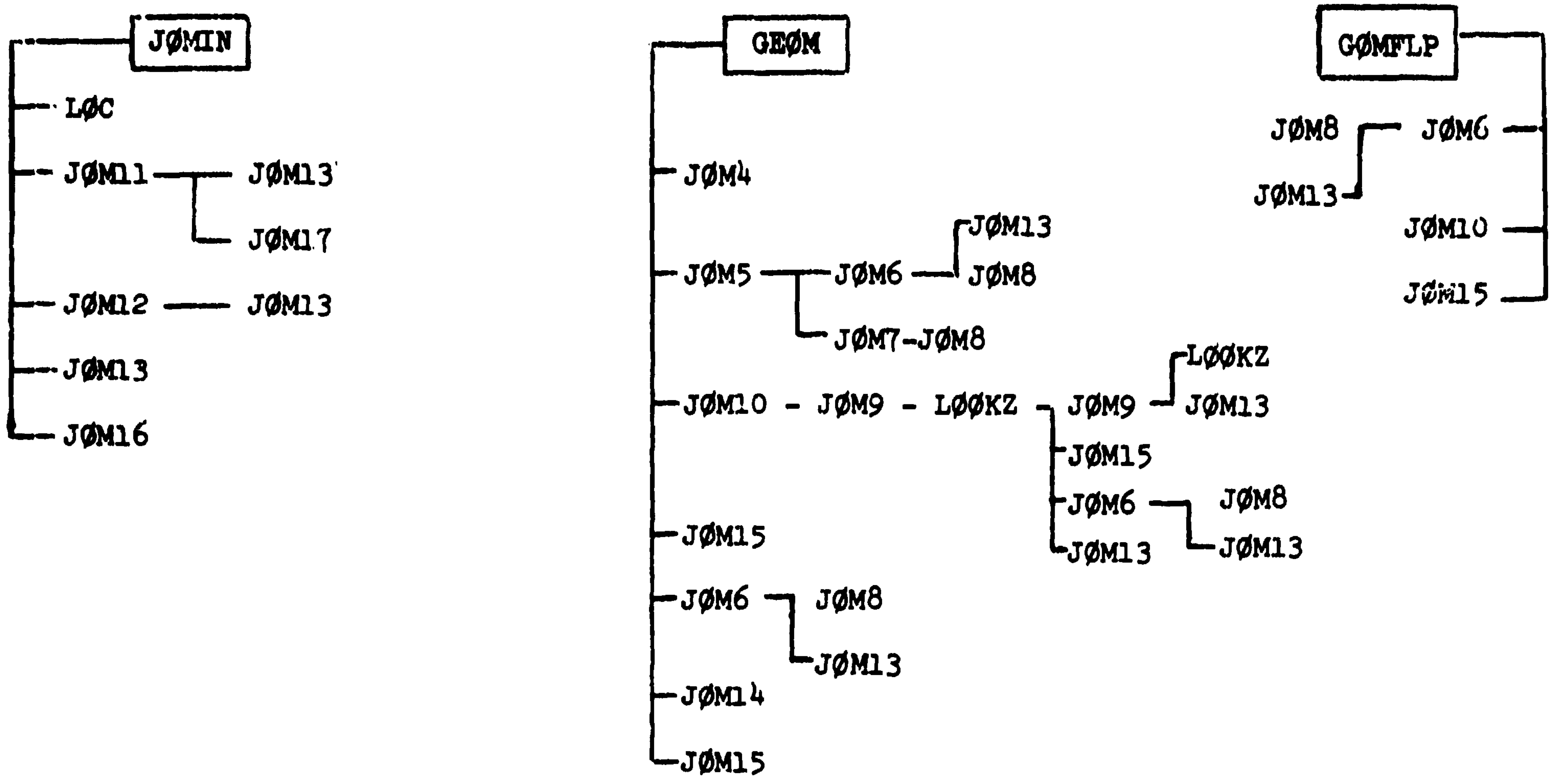


Fig. 4. Hierarchy of Subroutines in GENERAL GEØM

Changes to Geometry Packages

In order to implement the albedo option, it was necessary to make a minor change to the general GEØM package (and to all other special geometry packages). Previously a particle was always traced through the geometry to a collision point inside a medium. In the albedo option a particle is tracked to the boundary of the albedo medium where it undergoes a collision and departs in a different direction. In addition, the scattering and other routines needed to know the normal direction to the surface of the albedo medium and the region in which the albedo scattering occurred. To accomplish this, two subroutines were written and added to the GEØM package. These are GØMFLP, which prepares GEØM for the particle to reverse direction on its next flight, and NØRML, which calculates the normal to the albedo surface. One change was made to subroutine GEØM to implement this: at block boundary crossings the variable NCUE, indicating which boundary was crossed, is saved in NCUESV located in labelled common GEØM1. The storing of NCUESV is made at FØRTRAN statement 7 and the previous statement 7 becomes the next statement in the program.

In addition to the above modification, two other changes to the general GEØM package, which is described in reference 7, have been made. The first consists of putting several additional variables in labelled common for greater ease in examining dumps while debugging. The second change was made only to the IBM-360 version of GEØM. This involved changing the logical unit numbers used for the standard input and output units to variables NIN and NØUT which were stored in common JØMINX. The calling sequence for JØMIN was changed to CALL JØMIN (ADDR, NIN, NØUT) so that the user could convey the desired logical unit numbers to the GEØM subroutines.

Additional Parameters in Labelled Common

In JØM5 and JØM6, the GEØM56 common added the parameter REG which is a packed word that describes the present position of the particle with respect to the quadric surfaces in the block. A "1" indicates the particle is on the positive side of the surface, a "0" the negative side. The surfaces are in the order in which they were mentioned in the block description, starting at the last bit in the word and working back.

In JØM7 a new labelled common, GEØM70, was added to contain the variables P, Q, f(0), f(1), $(Q^2 - PF_o)$, u, v, w, Au, Bv, Cw,(Au + Dv + Ew), (Bv + Fw) used in calculating intersections with the quadric surfaces.

In JØM9 and LØØKZ, the parameters in their calling sequence were changed to X1, Y1, and Z1. Then the statements

XØNE = X1

YØNE = Y1

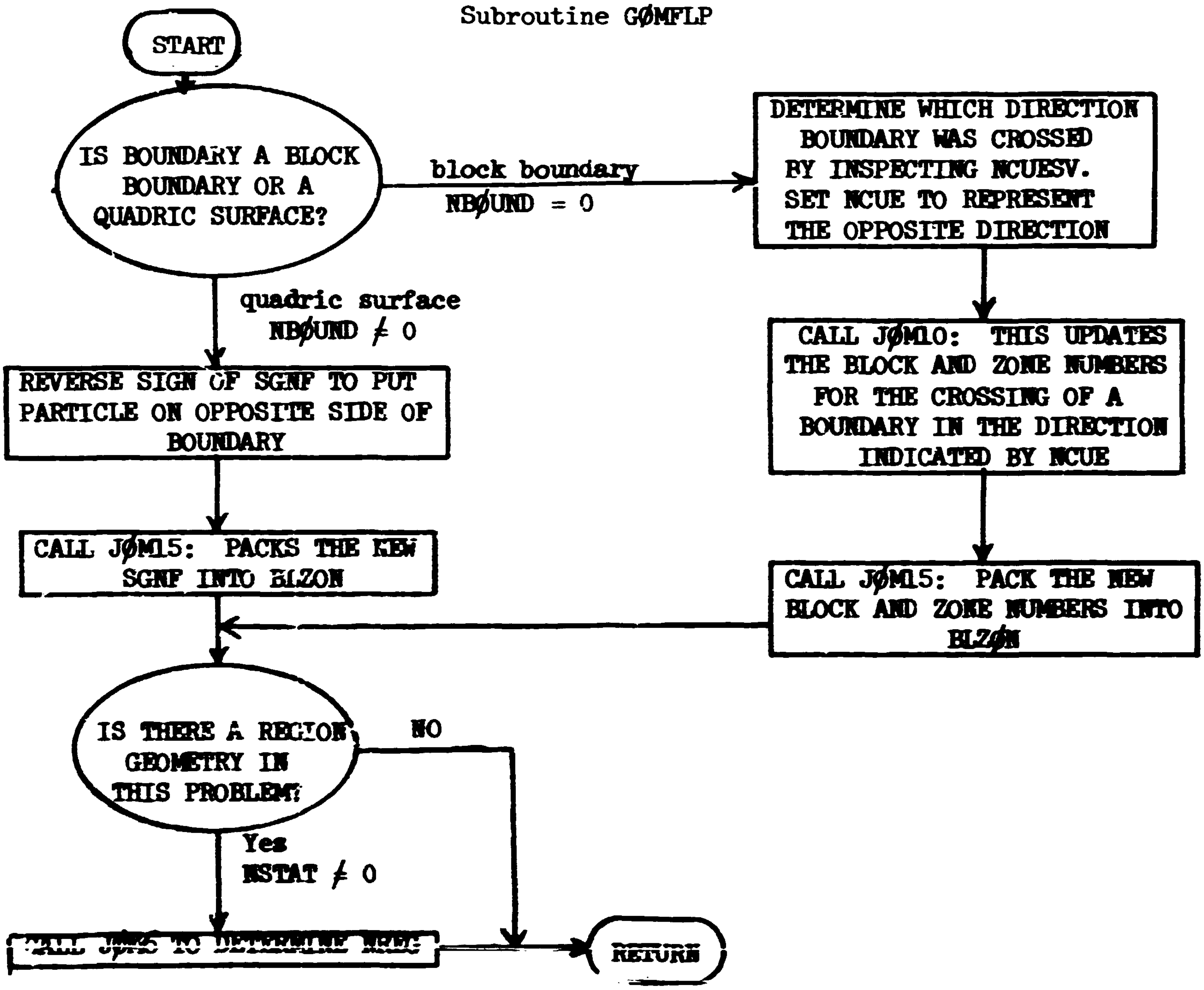
ZØNE = Z1

were added at the start of the program. Finally, XØNE, YØNE, and ZØNE were added to the labelled common GEØM39. The error message "YOU ARE LOST," indicating that a point is located outside the system, has been modified to print out the coordinates of the offending point.

Subroutine G∅MFLP (General GE∅M)

The purpose of this subroutine is to prepare GE∅M for the fact that an albedo-scattered particle is about to reverse direction while at a boundary. The indicators specifying that the particle has crossed the boundary and is entering the new medium must be flipped to indicate that the particle is reentering its original medium. It also calls J∅M6 to obtain the region number of the albedo-scattering site and stores this in NREG in GE∅MC common.

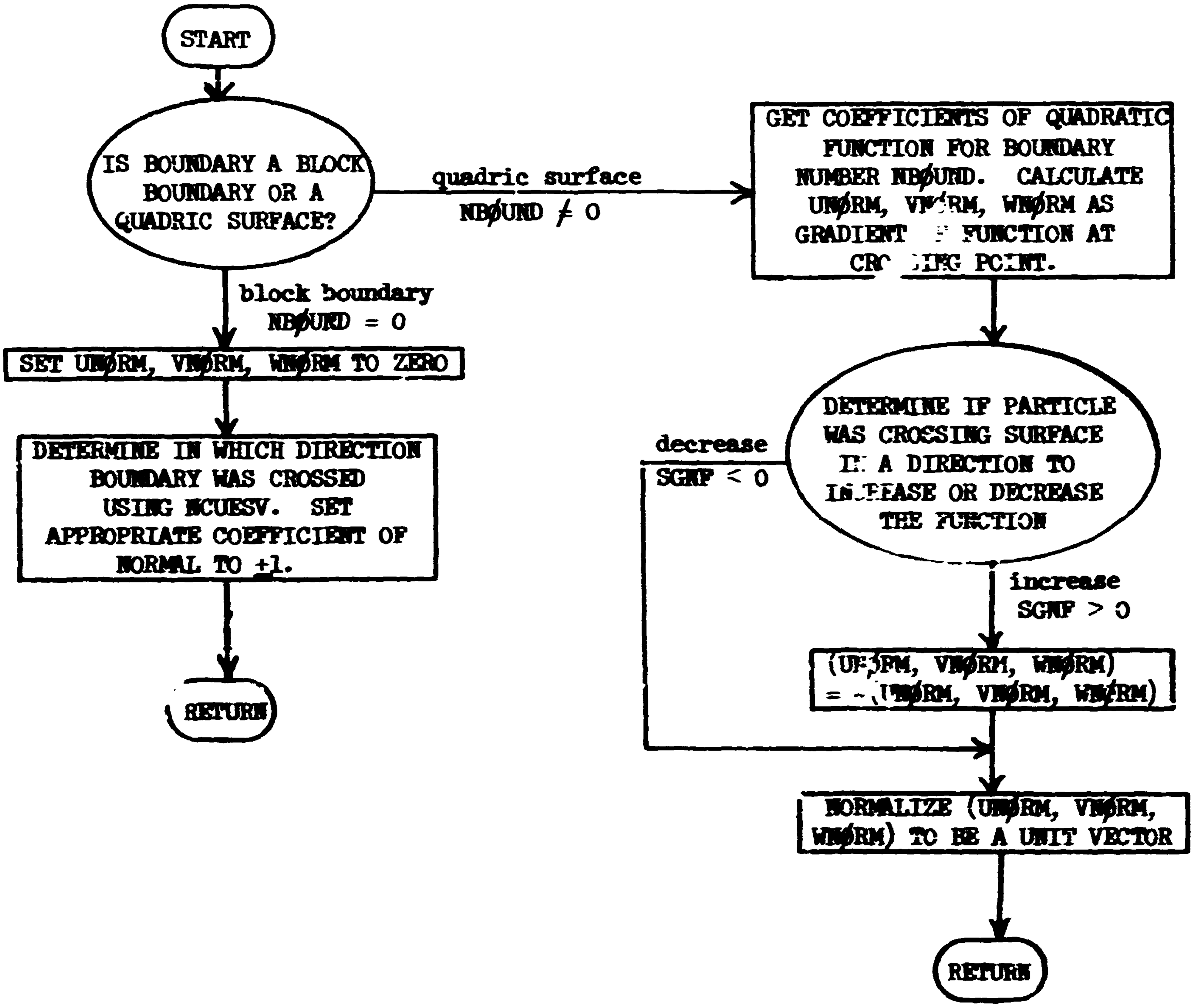
Subroutine G∅MFLP


START
IS BOUNDARY A BLOCK BOUNDARY OR A QUADRIC SURFACE?
block boundary
NB∅UND = 0
DETERMINE WHICH DIRECTION BOUNDARY WAS CROSSED BY INSPECTING NCUESV. SET NCUE TO REPRESENT THE OPPOSITE DIRECTION
quadric surface
NB∅UND ≠ 0
REVERSE SIGN OF SGNF TO PUT PARTICLE ON OPPOSITE SIDE OF BOUNDARY
CALL J∅M10: THIS UPDATES THE BLOCK AND ZONE NUMBERS FOR THE CROSSING OF A BOUNDARY IN THE DIRECTION INDICATED BY NCUE
CALL J∅M15: PACKS THE NEW SGNF INTO BLZON
CALL J∅M15: PACK THE NEW BLOCK AND ZONE NUMBERS INTO BLZ∅N
IS THERE A REGION GEOMETRY IN THIS PROBLEM?
NO
Yes
NSTAT ≠ 0
CALL J∅M6 TO DETERMINE NREG
RETURN

Subroutine NØRML (General GEØM)

Subroutine NØRML determines the normal to the albedo surface. The normal is stored in UNØRM, VNØRM, WNØRM in labelled common NØRMAL and always points out of the albedo medium.

Subroutine NØRML


START
IS BOUNDARY A BLOCK BOUNDARY OR A QUADRIC SURFACE?
quadric surface
NBØUND ≠ 0
GET COEFFICIENTS OF QUADRATIC FUNCTION FOR BOUNDARY NUMBER NBØUND. CALCULATE UNØRM, VNØRM, WNØRM AS GRADIENT OF FUNCTION AT CROSSING POINT.
block boundary
NBØUND = 0
SET UNØRM, VNØRM, WNØRM TO ZERO
DETERMINE IN WHICH DIRECTION BOUNDARY WAS CROSSED USING NCUESV. SET APPROPRIATE COEFFICIENT OF NORMAL TO ±1.
RETURN
DETERMINE IF PARTICLE WAS CROSSING SURFACE IN A DIRECTION TO INCREASE OR DECREASE THE FUNCTION
decrease
SGNF < 0
increase
SGNF > 0
(UNØRM, VNØRM, WNØRM) = -(UNØRM, VNØRM, WNØRM)
NORMALIZE (UNØRM, VNØRM, WNØRM) TO BE A UNIT VECTOR
RETURN

REFERENCES

1. V. R. Cain, "SAMBO, A Collision Analysis Package for Multigroup Monte Carlo Codes" (to be published).
2. Radiation Shielding Information Center, Oak Ridge National Laboratory, Oak Ridge, Tennessee.
3. 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).
4. W. W. Engle, Jr., "A User's Manual for ANISN," K-1693 (1967).
5. F. R. Mynatt, F. J. Muckenthaler, and P. N. Stevens, "Development of Two-Dimensional Discrete Ordinates Transport Theory for Radiation Shielding," CTC-INF-952 (1969).
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. D. C. Irving, V. R. Cain, and R. M. Freestone, Jr., "An Amplification of Selected Portions of the Ø5R Monte Carlo Code User's Manual," ORNL-TM-2601 (1969).
8. L. Cranberg <u>et al</u>., Phys. Rev. <u>103</u>, 662 (1956).
9. F. H. Clark and N. A. Betz, "Importance Sampling Devices for Selecting Track Lengths and Directions After Scatter in Ø5R," ORNL-TM-1484 (1966).
10. F. H. Clark, "The Exponential Transform as an Importance-Sampling Device - A Review," ORNL-RSIC-14 (1966).
11. 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).

12. T. J. Tyrrell, "TDUMP - A Translation Routine For Use in Dumping FORTRAN Arrays," ORNL-CF-70-7-8 (1970).

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

APPENDIX A

The 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:

$$\frac{1}{v}\frac{\partial}{\partial t}\,\phi(\bar{r},E,\bar{\Omega},t) + \nabla\cdot\phi(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\,\phi(\bar{r},E,\bar{\Omega},t)$$
$$= S(\bar{r},E,\bar{\Omega},t) + \int\int dE'd\bar{\Omega}'\,\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\,\phi(\bar{r},E',\bar{\Omega}',t) \qquad (1)$$

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.

$\left[\iiint\Sigma_s(\bar{r},E'\rightarrow E,\bar{\Omega}'\rightarrow\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'\right]dEd\bar{\Omega}$ = 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

$$\sum_{g=1}^{G}\Delta E_g = \int_0^{E_o} dE = E_o\text{, the maximum particle energy.}$$

A "group" form of Equation (1) is obtained by integrating each term with respect to the energy variable over the energy interval ΔE_g:

$$\frac{\partial}{\partial t}\int_{\Delta E_g}\frac{1}{v}\,\phi(\bar{r},E,\bar{\Omega},t)dE+\bar{\Omega}\cdot\nabla\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE+\int_{\Delta E_g}\Sigma_t(\bar{r},E)\phi(\bar{r},E,\bar{\Omega},t)dE$$

$$=\int_{\Delta E_g}S(\bar{r},E,\bar{\Omega},t)dE+\sum_{g'=g}^{1}\int_{\Delta E_{g'}}\int_{4\pi}dE'd\bar{\Omega}'\int_{\Delta E_g}\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\phi(\bar{r},E',\bar{\Omega}',t)dE \qquad (3)$$

Equation (3) provides the formal basis for the following group parameters:*

$\phi_g(\bar{r},\bar{\Omega},t)$ = time-dependent group angular flux,

$$=\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE\ , \qquad (4)$$

$\Sigma_t^g(\bar{r})$ = energy-averaged total cross section for the gth group,

$$\equiv\frac{\int_{\Delta E_g}\Sigma_t(\bar{r},E)\ \phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE} \qquad (5)$$

v_g = energy-averaged particle speed for the gth group,

$$\equiv\frac{\int_{\Delta E_g}\phi(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\frac{1}{v}\,\phi(\bar{r},E,\bar{\Omega},t)dE} \qquad (6)$$

$\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})$ = group g' to group g scattering cross section,

$$\equiv\frac{\int_{\Delta E_{g'}}\int_{\Delta E_g}\Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'dE}{\int_{\Delta E_{g'}}\phi(\bar{r},E',\bar{\Omega}',t)dE'} \qquad (7)$$

$S_g(\bar{r},\bar{\Omega},t)$ = distribution of source particles for the gth group,

$$\equiv\int_{\Delta E_g}S(\bar{r},E,\bar{\Omega},t)dE\ . \qquad (8)$$

*These parameters will be referred to as forward-weighted group parameters.

The group form of the Boltzmann equation expressed in terms of the aforedefined group parameters is given by

$$\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g(\bar{r},\bar{\Omega},t) + \bar{\Omega}\cdot\nabla\,\phi_g(\bar{r},\bar{\Omega},t) + \Sigma_t^g(\bar{r})\,\phi_g(\bar{r},\bar{\Omega},t)$$
$$= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\,\phi_{g'}(\bar{r},\bar{\Omega}',t)\ , \qquad (9)$$

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 in Fig. A.1.

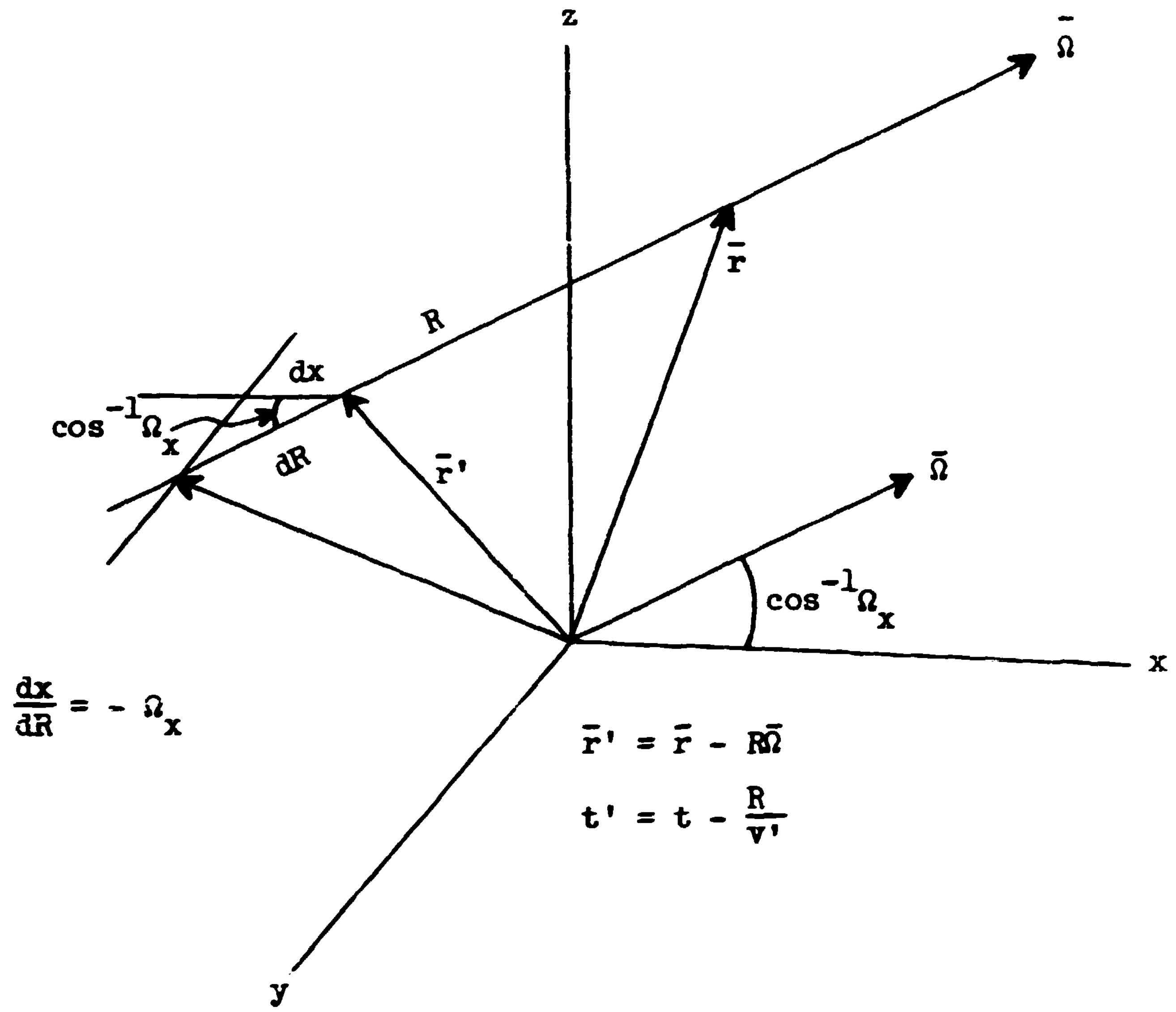


Figure A.1

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

$$\begin{aligned}\frac{d}{dR}\,\phi(\bar{r}',E,\bar{\Omega},t') &= -\,\Omega_x\frac{\partial\phi}{\partial x} - \Omega_y\frac{\partial\phi}{\partial y} - \Omega_z\frac{\partial\phi}{\partial z} - \frac{1}{v}\frac{\partial\phi}{\partial t}\\ &= -\,\bar{\Omega}\cdot\nabla\,\phi(\bar{r}',E,\bar{\Omega},t') - \frac{1}{v}\frac{\partial\phi}{\partial t}\ .\end{aligned} \qquad (10)$$

Equation (10) can be expressed in group notation as

$$-\frac{d}{dR}\,\phi_g(\bar{r}',\bar{\Omega},t') = \frac{1}{v_g}\frac{\partial}{\partial t}\,\phi_g(\bar{r}',\bar{\Omega},t') + \bar{\Omega}\cdot\nabla\,\phi_g(\bar{r}',\bar{\Omega},t')\ . \qquad (11)$$

Substitution of Eq. (11) into Eq. (8) with $\bar{r} \equiv \bar{r}'$ and $t \equiv t'$ yields

$$\begin{aligned}-\frac{d}{dR}\,\phi_g(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\,\phi_g(\bar{r}',\bar{\Omega},t') &= S_g(\bar{r}',\bar{\Omega},t)\\ &+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\,\Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\,\phi_{g'}(\bar{r}',\bar{\Omega},t')\ .\end{aligned} \qquad (12)$$

The integrating factor

$$e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

is introduced in the following manner:

$$\begin{aligned}\frac{d}{dR}\left[\phi_g(\bar{r}',\bar{\Omega},t')\,e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right] &= -\,e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\\ &\times\left\{-\frac{d\phi_g}{dR} + \Sigma_t^g(\bar{r}')\,\phi_g(\bar{r}',\bar{\Omega},t')\right\}\ .\end{aligned} \qquad (13)$$

Using Eq. (13), Eq. (12) can be rewritten as

$$-\frac{d}{dR}\left[\phi_g(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right] = e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} \qquad (14)$$

$$\times \left[S_g(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r}',\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega},t')\right].$$

Multiply Eq. (14) by dR and integrate (R = 0 to R = ∞); then

$$\phi_g(\bar{r},\bar{\Omega},t) - \phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}$$

$$= \int_0^\infty dR\, e^{-\int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'} \left[S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - \frac{R}{v})\right. \qquad (15)$$

$$\left. + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right]$$

Require that

$$\left[\phi_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'}\right] = 0\ , \qquad (16)$$

and introduce the "optical thickness"

$$\beta_g(\bar{r},R,\bar{\Omega}) \equiv \int_0^R \Sigma_t^g(\bar{r} - R'\bar{\Omega})dR'\ , \qquad (17)$$

and Eq. (15) becomes

$$\phi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta_g(\bar{r},R,\bar{\Omega})}\left[S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - R/v)\right. \qquad (18)$$

$$\left. + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\, \Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})\, \phi_{g'}(\bar{r}',\bar{\Omega}',t')\right].$$

Equation (18) will be referred to as the "Integral Flux Density Equation."

An effect of interest λ in group notation can be expressed as

$$\lambda_g = \iiint P_g^{\phi}(\bar{r},\bar{\Omega},t)\ \phi_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \quad , \tag{19}$$

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

$$\lambda = \sum_{g=1}^{G} \lambda_g \ . \tag{20}$$

<u>Integral Event Density Equation</u>

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 $\psi_g(\bar{r},\bar{\Omega},t)$:

$$\psi_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\ \Sigma_t^g(\bar{r})\ e^{-\beta_g(\bar{r},R,\bar{\Omega})} \left\{ S_g(\bar{r} - R\bar{\Omega},\bar{\Omega},t - R/v) \right.$$

$$\left. + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\ \frac{\Sigma_s^{g'\to g}(\bar{r} - R\bar{\Omega},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r}')}\ \psi_{g'}(\bar{r}',\bar{\Omega}',t') \right\} . \tag{22}$$

Equation (22) will be referred to as the "Integral Event Density Equation."

The effect oi 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),

$$\chi_g(\bar{r},\bar{\Omega},t) = S_g(\bar{r},\bar{\Omega},t) + \sum_{g'} \int_{4\pi} d\bar{\Omega}'\ \Sigma_s^{g'\to g}\ (\bar{r},\bar{\Omega}'\to\bar{\Omega})\ \phi_{g'}(\bar{r},\bar{\Omega}',t)\quad . \tag{25}$$

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):

$$\chi_g(\bar{r},\bar{\Omega},t)$$

$$= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\; \Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega}) \int_0^\infty dR\; e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t')$$

$$= S_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{1} \int_{4\pi} d\bar{\Omega}'\; \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})} \int_0^\infty dR\; \Sigma_t^{g'}(\bar{r})\; e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} \chi_{g'}(\bar{r}',\bar{\Omega}',t') . \tag{27}$$

The effect of interest λ_g can also be expressed in terms of the emergent particle density

$$\lambda_g = \iiint P_g^{\chi}(\bar{r},\bar{\Omega},t)\; \chi_g(\bar{r},\bar{\Omega},t) d\bar{r} d\bar{\Omega} dt \;. \tag{26}$$

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 $P_g^{\phi}(\bar{r}',\bar{\Omega},t')$. The sum of all such contributions to the effect of interest is given by

$$\int_0^\infty dR\; \Sigma_t^g(\bar{r}')\; e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\; P_g^{\phi}(\bar{r}',\bar{\Omega},t') \;,$$

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:

$$P_g^{\chi}(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\; \Sigma_t^g(\bar{r}')\; e^{-\beta^*(\bar{r},R,\bar{\Omega})} P_g^{\phi}(\bar{r}',\bar{\Omega},t') \;, \tag{29}$$

where

$P^X_g(\bar{r},\bar{\Omega},t)$ ≡ 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 $\beta_g(\bar{r},R,\bar{\Omega})$. $P^X_g(\bar{r},\bar{\Omega},t)$ can also be expressed in terms of $P^\dagger_g(\bar{r},\bar{\Omega},t)$ by substituting Eq. (25) into Eq. (29), yielding

$$P^X_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} P^\dagger_g(\bar{r}',\bar{\Omega},t') . \tag{31}$$

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), $[\Sigma_s^{g'\to g}(\bar{r},\Omega'\to\Omega)/\Sigma_s^{g'}(\bar{r})]$ 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

$$\psi_g(\bar{r},\Omega,t) = T_g(\bar{r}'\to\bar{r},\Omega)\ S_g(\bar{r}',\Omega,t') + C_{g'\to g}(\bar{r}',\Omega'\to\Omega)\psi_g(\bar{r}',\Omega',t') \ . \quad (36)$$

The term $T_g(\bar{r}',\bar{r},\Omega)S_g(\bar{r}',\Omega,t')$ can be identified as the "first collision source" and denoted by

$$S_c^g(\bar{r},\Omega,t) \equiv T_g(\bar{r}'\to\bar{r},\Omega)S_g(\bar{r}',\Omega,t') \ , \quad (37)$$

and the "Integral Event Density Equation" becomes

$$\psi_g(\bar{r},\Omega,t) = S_c^g(\bar{r},\Omega,t) + T(\bar{r}'\to\bar{r},\Omega)C_{g'\to g}(\bar{r}',\Omega'\to\Omega)\psi_g(\bar{r}',\Omega',t') \ . \quad (38)$$

Using the relationship $\psi_g(\bar{r},\Omega,t) = \Sigma_t^g(\bar{r})\phi_g(\bar{r},\Omega,t)$, Eq. (30) can be transformed into the "Integral Flux Density Equation:"

$$\phi_g(\bar{r},\Omega,t) = \frac{S_c^g(\bar{r},\Omega,t)}{\Sigma_t^g(\bar{r})} + T_g(\bar{r}'\to\bar{r},\Omega)C_{g'\to g}(\bar{r}',\Omega'\to\Omega)\ \frac{\Sigma_t^{g'}(\bar{r}')}{\Sigma_t^g(\bar{r})}\ \phi_g(\bar{r}',\Omega',t'). \quad (39)$$

Finally, the integral operators are introduced into Eq. (28) and the following form for the "Integral Emergent Particle Density Equation" is obtained:

$$\chi_g(\bar{r},\Omega,t) = S_g(\bar{r},\Omega,t) + C_{g'\to g}(\bar{r},\Omega'\to\Omega)\ T_{g'}(\bar{r}'\to\bar{r},\Omega')\ \chi_{g'}(\bar{r}',\Omega',t') \quad (40)$$

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

$$\chi_g(\bar{r},\bar{\Omega},t) = \sum_{n=0}^{\infty} \chi_g^n(\bar{r},\bar{\Omega},t) \quad , \qquad (41)$$

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 $\Sigma_t^{g_o}(\bar{r})e^{-\beta_{g_o}(\bar{r},R,\bar{\Omega}_o)}$ 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

$$\frac{\int_{4\pi} d\bar{\Omega}\ \Sigma_s^{g_0\to g}(\bar{r}_1,\bar{\Omega}_0\to\bar{\Omega})}{\Sigma_s^{g_0}(\bar{r}_1)} .$$

and then a new direction $\bar{\Omega}$ is determined from

$$\frac{\Sigma_s^{g_0\to g_1}(\bar{r}_1,\bar{\Omega}_0\to\bar{\Omega})}{\Sigma_s^{g_0\to g_1}(\bar{r}_1)} .$$

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 WATT and the estimator $P_g^X(\bar{r},\bar{\Omega},t)$.

<u>Derivation of the Adjoint Integro-Differential Boltzmann Transport Equation</u>

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:

$$\begin{aligned}&\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \frac{1}{v}\frac{\partial}{\partial t}\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\\ &\quad\times \nabla\cdot\bar{\Omega}\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E)\\ &\quad\times \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ S(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt\\ &\quad+ \iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt .\end{aligned} \qquad (42)$$

It can be shown that the following adjoint relationships are true for the conditions associated with a particle transport problem:

$$\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}\,\phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = -\iiiint \phi(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}$$

$$\times\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \left[\iiiint \frac{1}{v}\frac{\partial}{\partial t}\,\phi\ \phi^* d\bar{r}dEd\bar{\Omega}dt\right]_{\text{Boundary Term}} \qquad (43)$$

$$\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E)\ \phi(\bar{r},E,\bar{\Omega}t)d\bar{r}dEd\bar{\Omega}dt = \iiiint \phi(\bar{r},E,\bar{\Omega},t)$$

$$\times\ \Sigma_t(\bar{r},E)\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt\ , \qquad (44)$$

$$\iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\ \nabla\cdot\bar{\Omega}\ \phi(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = -\iiiint \phi(\bar{r},E,\bar{\Omega},t)$$

$$\times\ \nabla\cdot\bar{\Omega}\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \left[\iiiint \phi\ \phi^*\bar{\Omega}\cdot d\bar{S}dEd\bar{\Omega}dt\right]_{\text{Boundary Term}} \qquad (45)$$

$$\iiiint \phi^*(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E'\to E,\bar{\Omega}'\to\bar{\Omega})\ \phi(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt$$

$$= \iiiint \phi(\bar{r},E,\bar{\Omega},t) \iint \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\ \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt \qquad (46)$$

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

$$-\iiiint \phi(\bar{r},E,\bar{\Omega},t)\,\frac{1}{v}\frac{\partial}{\partial t}\,\phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt - \iiiint \phi(\bar{r},E,\bar{\Omega},t)$$

$$\times\ \nabla\cdot\bar{\Omega}\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi(\bar{r},E,\bar{\Omega},t)\ \Sigma_t(\bar{r},E)\ \phi^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt$$

$$= \iiiint \phi(\bar{r},E,\bar{\Omega},t)\ S^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt + \iiiint \phi(\bar{r},E,\bar{\Omega},t)$$

$$\times \iint \Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\ \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'d\bar{r}dEd\bar{\Omega}dt\ , \qquad (47)$$

where the adjoint source term $S^*(\bar{r},E,\bar{\Omega},t)$ is defined such that

$$\begin{aligned} &\iiiint \phi(\bar{r},E,\bar{\Omega},t)\; S^*(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt \\ &\quad = \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\; S(\bar{r},E,\bar{\Omega},t)\,d\bar{r}dEd\bar{\Omega}dt\ . \end{aligned} \tag{48}$$

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:

$$\begin{aligned} &\iiiint \phi(\bar{r},E,\bar{\Omega},t) \Bigg[-\frac{1}{v}\frac{\partial}{\partial t}\, \phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\; \phi^*(\bar{r},E,\bar{\Omega},t) \\ &\quad + \Sigma_t(\bar{r},E)\; \phi^*(\bar{r},E,\bar{\Omega},t) - S^*(\bar{r},E,\bar{\Omega},t) - \iint \Sigma_s(\bar{r},E\!\rightarrow\!E',\bar{\Omega}\!\rightarrow\!\bar{\Omega}') \\ &\quad \times\, \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}' \Bigg]\, d\bar{r}dEd\bar{\Omega}dt = 0\ . \end{aligned} \tag{49}$$

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:

$$\begin{aligned} &\Bigg[-\frac{1}{v}\frac{\partial}{\partial t}\, \phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\phi^*(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\phi^*(\bar{r},E,\bar{\Omega},t) - S^*(\bar{r},E,\bar{\Omega},t) \\ &\quad - \iint \Sigma_s(\bar{r},E\!\rightarrow\!E',\bar{\Omega}\!\rightarrow\!\bar{\Omega}')\; \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}' \Bigg] = 0\ . \end{aligned}$$

This condition also satisfies Eq. (49) exactly and provides the following $\phi^*(\bar{r},E,\bar{\Omega},t)$-defining integro-differential equation:

$$\begin{aligned} &-\frac{1}{v}\frac{\partial}{\partial t}\, \phi^*(\bar{r},E,\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\; \phi^*(\bar{r},E,\bar{\Omega},t) + \Sigma_t(\bar{r},E)\; \phi^*(\bar{r},E,\bar{\Omega},t) \\ &\quad = S^*(\bar{r},E,\bar{\Omega},t) + \iint \Sigma_s(\bar{r},E\!\rightarrow\!E',\bar{\Omega}\!\rightarrow\!\bar{\Omega}')\; \phi^*(\bar{r},E',\bar{\Omega}',t)dE'd\bar{\Omega}'\ , \end{aligned} \tag{50}$$

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:

$$-\frac{1}{\hat{v}_g}\frac{\partial}{\partial t}\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) - \nabla\cdot\bar{\Omega}\,\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) + \hat{\Sigma}^g_t(\bar{r})\,\hat{\phi}^*_g(\bar{r},E,\bar{\Omega},t)$$
$$= S^*_g(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{G}\int d\bar{\Omega}'\,\hat{\Sigma}_s^{g\to g'}(\bar{\Omega}\to\bar{\Omega}')\,\hat{\phi}^*_{g'}(\bar{r},\bar{\Omega}',t)\,, \qquad (51)$$
$$g = 1,2,\ldots G$$

where

$$\hat{\phi}^*_g(\bar{r},\bar{\Omega},t) = \frac{1}{\Delta E_g}\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE\,, \qquad (52)$$

$$\hat{v}_g = \frac{\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\frac{1}{v}\,\phi^*(\bar{r},E,\bar{\Omega},t)dE} \qquad (53)$$

$$\hat{\Sigma}^g_t(\bar{r}) \equiv \frac{\int_{\Delta E_g}\Sigma_t(\bar{r},E)\,\phi^*(\bar{r},E,\bar{\Omega},t)dE}{\int_{\Delta E_g}\phi^*(\bar{r},E,\bar{\Omega},t)dE}\,, \qquad (54)$$

$$\hat{\Sigma}_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}') \equiv \frac{\int_{\Delta E_g}\int_{\Delta E_{g'}}\Sigma_s(\bar{r},E\to E',\bar{\Omega}\to\bar{\Omega}')\,\phi^*(\bar{r},E',\bar{\Omega}',t)dE'dE}{\int_{\Delta E_{g'}}\phi^*(\bar{r},E',\bar{\Omega}',t)dE'}\,, \qquad (55)$$

$$\hat{S}^*_g(\bar{r},\bar{\Omega},t) = \frac{1}{\Delta E_g}\int_{\Delta E_g}S^*(\bar{r},E,\bar{\Omega},t)dE\,. \qquad (56)$$

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

$$\begin{aligned} -\frac{1}{v_g}\frac{\partial}{\partial t}\phi_g^*(\bar{r},\bar{\Omega},t) - \Delta\cdot\bar{\Omega}\phi_g^*(\bar{r},\bar{\Omega},t) + \Sigma_t^g(\bar{r})\phi_g^*(\bar{r},E,\bar{\Omega},t) \\ = S_g^*(\bar{r},\bar{\Omega},t) + \sum_{g'=g}^{G}\int d\bar{\Omega}'\;\Sigma_s^{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\phi_g^*(\bar{r},\bar{\Omega}',t)\;, \\ g = 1,2,\ldots G. \end{aligned} \tag{57}$$

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.,

$$\phi_g^*(\bar{r},\bar{\Omega},t) \neq \frac{1}{\Delta E_g}\int_{\Delta E_g} \phi^*(\bar{r},E,\bar{\Omega},t)dE \quad .$$

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 weignted 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 in Fig. A.2, 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

$$\frac{d}{dR}\,\phi_g^*(r',\Omega,t') = \frac{1}{v_g}\frac{\partial}{\partial t}\,\phi_g^*(r',\Omega,t') + \nabla\cdot\bar{\Omega}\,\phi_g^*(r',\Omega,t') \;. \qquad (58)$$

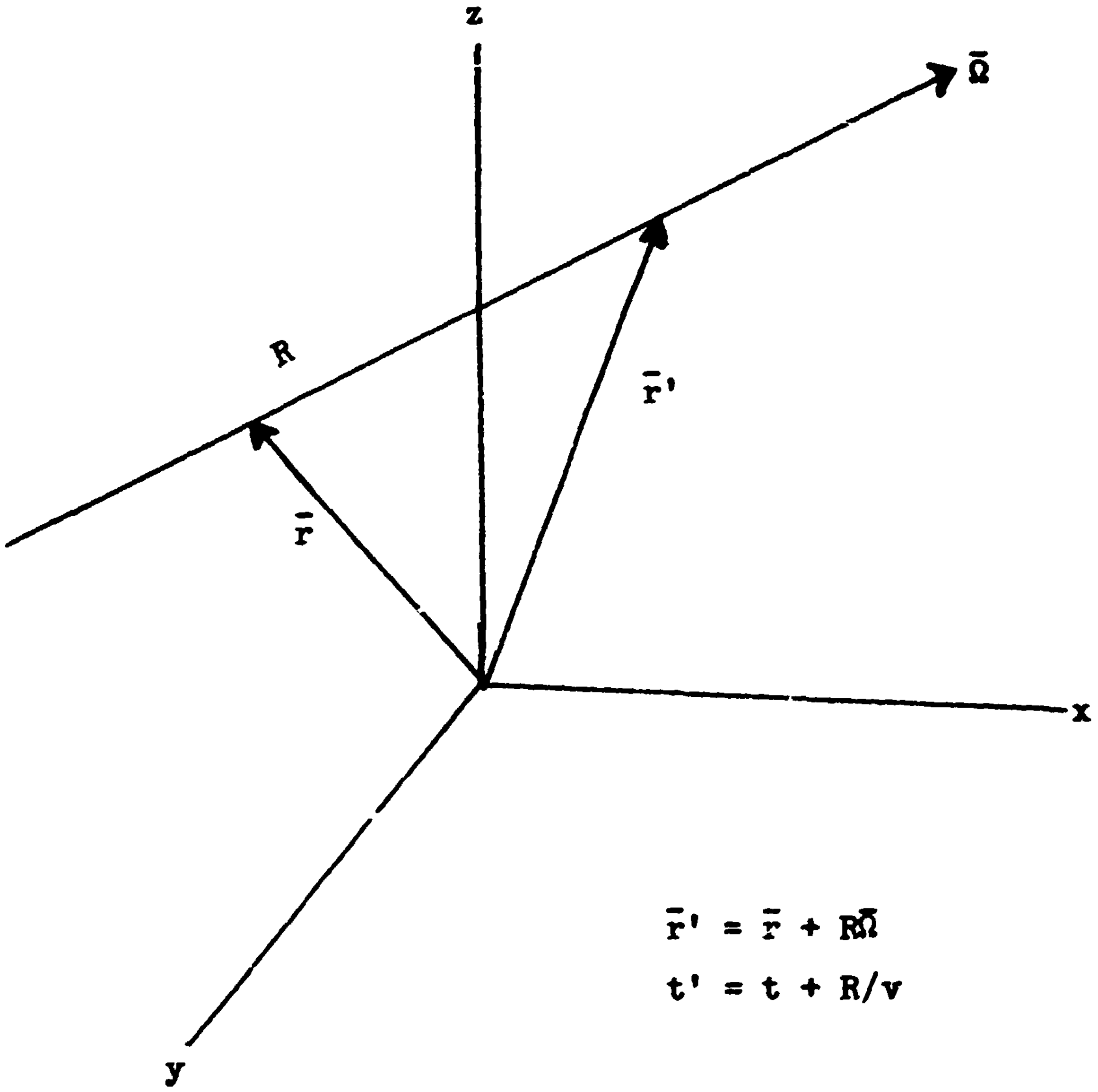


Figure A.2

Use of the integrating factor $e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}$ provides the following relationship:

$$\frac{d}{dR}\left\{\phi_g^*(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\right\}$$

$$= \frac{d\phi_g^*}{dR}(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} - \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')$$

$$\times\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} = e^{-\int_0^R \Sigma_t^g(\bar{r}' + R'\bar{\Omega})dR'} \tag{59}$$

$$\times \left[\frac{d}{dR}\,\phi_g^*(\bar{r}',\bar{\Omega},t') - \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')\right] .$$

Equation (59), together with Eq. (58), can be arranged to give

$$\left[-\frac{1}{v_g}\frac{\partial}{\partial t}\,\phi_g^*(\bar{r}',\bar{\Omega},t') - \nabla\cdot\bar{\Omega}\,\phi_g^*(\bar{r}',\bar{\Omega},t') + \Sigma_t^g(\bar{r}')\,\phi_g^*(\bar{r}',\bar{\Omega},t')\right]$$

$$= e^{+\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\;\frac{d}{dR}\left[\phi_g^*(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\right]. \tag{60}$$

It is noted that Eq. (60) is identically the left-hand side of Eq. (57) which can now be rewritten as

$$-\frac{d}{dR}\left[\phi_g^*(\bar{r}',\bar{\Omega},t')\, e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\right] = e^{-\int_0^R \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'} \tag{61}$$

$$\times \left\{S_g^*(\bar{r}',\bar{\Omega},t') + \sum_{g'=g}^{G} \int d\bar{\Omega}'\; \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\;\phi_{g'}^*(\bar{r}',\bar{\Omega}',t')\right\}.$$

Integrate Eq. (61) from R = 0 to R = ∞ and assume that

$$\{\phi^*_g(\infty,\bar{\Omega},t_\infty)e^{-\int_0^\infty \Sigma_t^g(\bar{r} + R'\bar{\Omega})dR'}\} \equiv 0 \; ; \qquad (62)$$

then the following integral expression for $\phi^*_g(\bar{r},\bar{\Omega},t)$ is obtained:

$$\phi^*_g(\bar{r},\bar{\Omega},t) = \int_0^\infty dR\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})}\{S^*_g(\bar{r} + R\bar{\Omega},\bar{\Omega},t + R/v_g) \qquad (63)$$

$$+ \sum_{g'} \int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r} + R\bar{\Omega},\bar{\Omega}\to\bar{\Omega}')\, \phi^*_{g'}(\bar{r}',\bar{\Omega}',t')\} .$$

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

$$S^*_{Tg}(\bar{r},\bar{\Omega},t) = \int_0^R dR\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})}\, S^*_g(\bar{r} + R\bar{\Omega},\bar{\Omega},t + R/v_g) , \qquad (64)$$

and Eq. (63) can be rewritten as

$$\phi^*_g(\bar{r},\bar{\Omega},t) = S^*_{Tg}(\bar{r},\bar{\Omega},t) + \int dR\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} \sum_{g'} \int d\bar{\Omega}'\, \Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')$$

$$\times\, \phi^*_{g'}(\bar{r}',\bar{\Omega}',t') = S^*_{Tg}(\bar{r},\bar{\Omega},t) + \int_0^\infty dR\, \Sigma_t^g(\bar{r} + R\bar{\Omega})\, e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} \qquad (65)$$

$$\times \sum_{g'} \int d\bar{\Omega}'\, \frac{\Sigma_s^{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma_t^g(\bar{r}')}\, \phi^*_{g'}(\bar{r}',\bar{\Omega}',t') ,$$

and in terms of the transport and collision operators, Eq. (65) becomes

$$\phi^*_g(\bar{r},\bar{\Omega},t) = S^*_{Tg}(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\; C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\; \phi^*_{g'}(\bar{r}',\bar{\Omega}',t') \; . \tag{66}$$

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

$$\chi^*_g(\bar{r},\bar{\Omega},t) = S^*_{Tg}(\bar{r},\bar{\Omega},t) + T_g(\bar{r}\to\bar{r}',\bar{\Omega})\; C_{g\to g'}(\bar{\Omega}\to\bar{\Omega}',\bar{r}')\; \chi^*_{g'}(\bar{r}',\bar{\Omega}',t') \; . \tag{67}$$

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

$$\iiiint \phi(\bar{r},E,\bar{\Omega},t)\; S^*(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = \iiiint \phi(\bar{r},E,\bar{\Omega},t) \times P^{\phi}(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt = \lambda \; . \tag{68}$$

According to Eq. (48), the effect of interest λ would also be given by

$$\lambda = \iiiint \phi^*(\bar{r},E,\bar{\Omega},t)\; S(\bar{r},E,\bar{\Omega},t)d\bar{r}dEd\bar{\Omega}dt \; . \tag{69}$$

The effect of interest as given by Eq. (69) can also be expressed in group notation

$$\lambda_g = \iiint \hat{\phi}^*_g(\bar{r},\bar{\Omega},t)\hat{S}_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt$$

$$= \iiint \phi_g(\bar{r},\bar{\Omega},t)\hat{S}^*_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \; ,$$

where

$\hat{\phi}^*_g(\bar{r},\bar{\Omega},t)$ is the group adjoint flux corresponding to the adjoint weighted group parameters,

$$\hat{S}_g(\bar{r},\bar{\Omega},t) = \Delta E_g \frac{\int\limits_{\Delta E_g} \phi^*(\bar{r},E,\bar{\Omega},t)S(\bar{r},E,\bar{\Omega},t)dE}{\hat{\phi}^*_g(\bar{r},\bar{\Omega},t)}$$

$$\hat{S}^*_g(\bar{r},\bar{\Omega},t) = \frac{\int\limits_{\Delta E_g} S^*(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)dE}{\phi_g(\bar{r},\bar{\Omega},t)}$$

$$= \frac{\int\limits_{\Delta E_g} P^{\hat{\phi}}(\bar{r},E,\bar{\Omega},t)\phi(\bar{r},E,\bar{\Omega},t)dE}{\phi_g(\bar{r},\bar{\Omega},t)} = P^{\phi}_g(\bar{r},\bar{\Omega},t) \ .$$

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

$$\begin{aligned}\lambda_g &= \iiint \phi^*_g(\bar{r},\bar{\Omega},t)S_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \\ &= \iiint \phi_g(\bar{r},\bar{\Omega},t)S^*_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \ ,\end{aligned} \tag{70}$$

where

$\phi^*(\bar{r},\bar{\Omega},t)$ is the group adjoint flux corresponding to the forward weighted group parameters,

$$\begin{aligned}S_g(\bar{r},\bar{\Omega},t) &= \int\limits_{\Delta E_g} S(\bar{r},E,\bar{\Omega},t)dE \ , \\ S^*_g(\bar{r},\bar{\Omega},t) &= P^{\phi}_g(\bar{r},\bar{\Omega},t) \ .\end{aligned} \tag{71}$$

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

$$S^{*}_{Tg}(\bar{r},\bar{\Omega},t) = \int dR\, e^{-\beta^{*}(\bar{r},R,\bar{\Omega})}\, P^{\phi}_{g}(\bar{r}',\bar{\Omega},t') \,, \tag{72}$$

and according to Equations (25) and (29), Eq. (72) can be rewritten as Equations (73) and (74), respectively:

$$S^{*}_{Tg}(\bar{r},\bar{\Omega},t) = \int dR\, \Sigma^{g}_{t}(\bar{r}')\, e^{-\beta^{*}_{g}(\bar{r},R,\bar{\Omega})}\, P^{\phi}_{g}(\bar{r}',\bar{\Omega},t') \tag{73}$$

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:"

$$\chi^{*}_{g}(\bar{r},\bar{\Omega},t) = T_{g}(\bar{r}\to\bar{r}',\bar{\Omega})\, \{P^{\phi}_{g}(\bar{r}',\bar{\Omega},t') + C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')\, \chi^{*}_{g'}(\bar{r}',\bar{\Omega}',t')\} \tag{75}$$

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,

$$\chi_g^*(\bar{r},\bar{\Omega},t) = \int_0^\infty dP\ \Sigma_t^g(\bar{r}')\ e^{-\beta_g^*(\bar{r},R,\bar{\Omega})}\ W_g(\bar{r}',\bar{\Omega},t') \tag{77}$$

or

$$\chi_g^*(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}',\bar{\Omega})\ W_g(\bar{r}',\bar{\Omega},t')\ . \tag{78}$$

A comparison of Eq. (78) with Eq. (75) would show that $W_g(\bar{r},\bar{\Omega},t)$ can be identified as

$$W_g(\bar{r},\bar{\Omega},t) = P_g^\psi(\bar{r},\bar{\Omega},t) + C_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ \chi_{g'}^*(\bar{r},\bar{\Omega}',t)\ , \tag{79}$$

and substitution of Eq. (78) into Eq. (79) yields the defining equation for the "Event-Value Function"

$$W_g(\bar{r},\bar{\Omega},t) = P_g^\psi(\bar{r},\bar{\Omega},t) + C_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ T_{g'}(\bar{r}\to\bar{r}',\bar{\Omega}')\ W_{g'}(\bar{r}',\bar{\Omega}',t')\ . \tag{80}$$

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 λ_g is given by

$$\lambda_g = \iiint S_c^g(\bar{r},\bar{\Omega},t)\ W_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt\ . \tag{81}$$

Integral Emergent Adjuncton Density Equation

The solution of either the point-value equation, Eq. (76), or the event-value equation, Eq. (79), 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.

5

$$\chi^*_g(\bar{r},\bar{\Omega},t) = P^{\psi}_g(\bar{r},\bar{\Omega},t) + \int dR \; \Sigma^g_t(\bar{r}) \; e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} \left(\frac{\Sigma^g_t(\bar{r}')}{\Sigma^g_t(\bar{r})}\right) \tag{82}$$

$$\times \sum_{g'} \int d\Omega' \; \frac{\Sigma^{g\to g'}_s(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\sum\limits_{g} \Sigma^{g\to g'}_s(\bar{r}',\bar{\Omega}\to\bar{\Omega}')} \left(\frac{\sum\limits_{g} \Sigma^{g\to g'}_s(\bar{r}',\bar{\Omega}\to\bar{\Omega}')}{\Sigma^g_t(\bar{r}')}\right) \chi^*_g(\bar{r}',\bar{\Omega}',t') \; .$$

The additional weight factor $[\Sigma^g_t(\bar{r}')/\Sigma^g_t(\bar{r})]$ arises since Eq. (76) and its altered form (Eq. (82), are actually flux-like 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:

$$H_g(\bar{r},\bar{\Omega},t) \equiv \Sigma^g_t(\bar{r}) \; \chi^*_g(\bar{r},\bar{\Omega},t) \tag{83}$$

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):

$$\chi^*_g(\bar{r},\bar{\Omega},t) = \int dR \; \Sigma^g_t(\bar{r}') \; e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} [P^{\psi}_g(\bar{r}',\bar{\Omega},t') \tag{85}$$

$$+ C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \; \chi_{g'}(\bar{r}',\bar{\Omega}',t')] \; .$$

Multiply Eq. (85) by $\Sigma^g_t(\bar{r})$ and rearrange as follows:

$$\Sigma^g_t(\bar{r}) \; \chi^*_g(\bar{r},\bar{\Omega},t) = \int dR \; \Sigma^g_t(\bar{r}) \; e^{-\beta^*_g(\bar{r},R,\bar{\Omega})} [\Sigma^g_t(\bar{r}') \; P^{\psi}_g(\bar{r}',\bar{\Omega},t') \tag{86}$$

$$+ \check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \; \Sigma^{g'}_t(\bar{r}') \; \chi^*_{g'}(\bar{r}',\bar{\Omega}',t')]$$

where

$$\check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \equiv \frac{\Sigma_t^g(\bar{r}')}{\Sigma_t^{g'}(\bar{r})} C_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}') \ . \qquad (87)$$

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

$$H_g(\bar{r},\bar{\Omega},t) = T_g(\bar{r}\to\bar{r}\ ,\bar{\Omega})[P_g^\phi(\bar{r}',\bar{\Omega},t') + \check{C}_{g\to g'}(\bar{r}',\bar{\Omega}\to\bar{\Omega}')H_{g'}(\bar{r}',\bar{\Omega}',t')] \ . \qquad (88)$$

A comparison of Eq. (88) with Eq. (84) reveals that

$$G_g(\bar{r},\bar{\Omega},t) = P_g^\phi(\bar{r},\bar{\Omega},t) + \check{C}_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ H_{g'}(\bar{r},\bar{\Omega}',t) \ , \qquad (89)$$

and the subsequent substitution of Eq. (84) into Eq. (89) yields the following defining equation for the adjoint emergent particle density:

$$G_g(\bar{r},\bar{\Omega},t) = P_g^\phi(\bar{r},\bar{\Omega},t) + \check{C}_{g\to g'}(\bar{r},\bar{\Omega}\to\bar{\Omega}')\ T_{g'}(\bar{r}\to\bar{r}',\bar{\Omega}')\ G_{g'}(\bar{r}',\bar{\Omega}',t'). \qquad (90)$$

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 (P'→P) 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, $\hat{\Omega} = -\Omega$) 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 density 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:

$$\lambda_g = \iiint P_g^\phi(\bar{r},\bar{\Omega},t)\ \phi_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{95}$$

$$= \iiint \frac{P_g^\phi(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\rightarrow\bar{r},\bar{\Omega})\ \chi_g(\bar{r}',\bar{\Omega},t')d\bar{r}d\bar{\Omega}dt$$

$$\lambda_g = \iiint S_g(\bar{r},\bar{\Omega},t)\chi_g^*(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint S_g(\bar{r},\bar{\Omega},t)\chi_g^*(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{96}$$

$$\lambda_g = \iiint S_c^g(\bar{r},\bar{\Omega},t)W_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint S_c^g(\bar{r},\bar{\Omega},t)W_g(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{97}$$

$$\lambda_g = \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} H_g(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt = \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} H_g(\bar{r},-\hat{\Omega},t)d\bar{r}d\bar{\Omega}dt \tag{98}$$

$$\lambda_g = \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\rightarrow\bar{r},\bar{\Omega})G_g(\bar{r}',\bar{\Omega},t')d\bar{r}d\bar{\Omega}dt$$

$$= \iiint \frac{S_g(\bar{r},\bar{\Omega},t)}{\Sigma_t^g(\bar{r})} T_g(\bar{r}'\rightarrow\bar{r},-\hat{\Omega})\ G_g(\bar{r}',-\hat{\Omega},t')d\bar{r}d\bar{\Omega}dt\ . \tag{99}$$

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

$$S_g^n(\bar{r},\bar{\Omega},t) = \int_{\Delta E_g} S^n(\bar{r},E,\bar{\Omega},t)dE \tag{100}$$

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 $\bar{\Omega}$,

$$S^n(\bar{r},E,\bar{\Omega},t) = \frac{f(E)}{4\pi} \iint_{4\pi} dE'd\bar{\Omega}' \; \nu\Sigma_f(\bar{r},E')\phi^{n-1}(\bar{r},E',\bar{\Omega}',t) \tag{101}$$

$f(E)dE$ = 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

$$S_g^n(\bar{r},\bar{\Omega},t) = \frac{f_g}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \; \nu\Sigma_f^{g'}(\bar{r}) \; \phi_{g'}^{n-1}(\bar{r},\bar{\Omega}',t) \; , \tag{102}$$

[*] 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):

$$\chi_g^n(\bar{r},\bar{\Omega},t) = S_g^n(\bar{r},\bar{\Omega},t)$$

$$+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})}\int_0^\infty dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\chi_{g'}^n(\bar{r}',\bar{\Omega}',t')$$

$$= S_g^n(\bar{r},\bar{\Omega},t) + C_{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})\ T_{g'}(\bar{r}'\to\bar{r},\bar{\Omega}')\ \chi_{g'}^n(\bar{r}',\bar{\Omega}',t')\ . \tag{107}$$

Equations (105) and (107) can be combined and written as an eigenvalue equation in seven-dimensional phase space

$$\chi_g(\bar{r},\bar{\Omega},t) = \frac{1}{k}\frac{f_g}{4\pi}\sum_{g'=G}^{1}\int_{4\pi} d\bar{\Omega}'\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})} dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_g(\bar{r},R,\bar{\Omega})}\ \chi_g(\bar{r}',\bar{\Omega},t')$$

$$+ \sum_{g'=g}^{1}\int_{4\pi} d\bar{\Omega}'\ \frac{\Sigma_s^{g'\to g}(\bar{r},\bar{\Omega}'\to\bar{\Omega})}{\Sigma_t^{g'}(\bar{r})}\int_0^{\infty} dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\ \chi_{g'}(\bar{r}',\bar{\Omega}',t'). \tag{108}$$

The usual objective in a reactor calculation is to find the eigenfunctions $\chi_g(\bar{r},\bar{\Omega},t)$, and the eigenvalue k. In MORSE 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

$$k_n = \frac{\displaystyle\sum_{g'=G}^{1}\iiint d\bar{r}d\bar{\Omega}'dt\frac{\nu\Sigma_f^{g'}(\bar{r})}{\Sigma_t^{g'}(\bar{r})}\int_0^{\infty} dR\ \Sigma_t^{g'}(\bar{r})\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')}\ \chi_{g'}^{n}(\bar{r}',\bar{\Omega}',t')}{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^{n}\ d\bar{r}d\bar{\Omega}dt} \tag{109}$$

which can also be expressed as the ratio of successive sources

$$k_n = \frac{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^{n+1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt}{\displaystyle\sum_{g'=G}^{1}\iiint S_{g'}^{n}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt}\ . \tag{110}$$

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:

$$\text{FWLØ}_{n+1} = \text{FWLØ}_n \cdot \bar{k}_n \cdot \frac{\text{(fission neutrons produced during the nth batch)}}{\text{(source neutrons introduced into the first batch)}} , \quad (111)$$

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

$$\begin{aligned}\lambda_g^{n-1} &= \iiint [P_g^\phi(\bar{r},\bar{\Omega},t)]^n \phi_g^{n-1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \\ &= \iiint \left\{ \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi} \chi_{g'}^{*n-1}(\bar{r},\bar{\Omega}',t) \right\} \phi_g^{n-1}(\bar{r},\bar{\Omega},t)d\bar{r}d\bar{\Omega}dt \end{aligned} \tag{113}$$

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

$$\begin{aligned}[P_g^\phi(\bar{r},\bar{\Omega},t)]^n &= \nu\Sigma_f^g(\bar{r}) \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}' \frac{f_{g'}}{4\pi\ \Sigma_t^{g'}(\bar{r})} T_{g'}(\bar{r}'\mapsto\bar{r},\bar{\Omega}')G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') \\ &= \frac{\nu\Sigma_f^g(\bar{r})}{4\pi} \sum_{g'=G}^{1} \int_{4\pi} d\bar{\Omega}'\ f_{g'} \int_0^\infty dR\ e^{-\beta_{g'}(\bar{r},R,\bar{\Omega}')} G_{g'}^{n-1}(\bar{r}',\bar{\Omega}',t') \ . \end{aligned} \tag{116}$$

Noting that the fission process is independent of the incident neutron's direction and that the fission neutrons are emitted isotropically

$$[P^{\phi}_{g}(\bar{r},\bar{\Omega},t)]^{n} \equiv [P^{\phi}_{g}(\bar{r},\hat{\Omega},t)]^{n} \quad , \tag{117}$$

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

$$[P^{\phi}_{g}(\bar{r},\hat{\Omega},t)]^{n} = \frac{1}{4\pi}\,\frac{\nu\Sigma^{g}_{f}(\bar{r})}{\nu\Sigma_{f}(\bar{r})} \sum_{g'=G}^{1} \int_{4\pi} d\hat{\Omega}'\ [f_{g'}\ \nu\Sigma_{f}(\bar{r})] \int_{0}^{\infty} dR\ e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')} \times G^{n-1}_{g'}(\bar{r}',\hat{\Omega}',t') \quad , \tag{118}$$

where

$$\nu\Sigma_{f}(\bar{r}) = \sum_{g=G}^{1} \nu\Sigma^{g}_{f}(\bar{r}), \tag{119}$$

$\dfrac{\nu\Sigma^{g}_{f}(\bar{r})}{\nu\Sigma_{f}(\bar{r})}$ = 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 charged their roles. The adjoint-fission group cross section is $[f_{g}\cdot\nu\Sigma_{f}(\bar{r})]$ and the energy distribution of the adjunctons emerging from an adjoint fission is $\nu\Sigma^{g}_{f}(\bar{r})/\nu\Sigma_{f}(\bar{r})$.

*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:

$$k_n = \frac{\sum\limits_{g'=G}^{1}\iiint d\bar{r}\,d\hat{\Omega}'\,dt\,[f_{g'}\cdot\nu\Sigma_f^{g'}(\bar{r})]\int_0 dR\; e^{-\beta_{g'}(\bar{r},R,\hat{\Omega}')}\;G_{g'}^{n}(\bar{r}',\hat{\Omega}',t')}{\sum\limits_{g'=G}^{1}\iiint [P_{g'}^{\phi}(\bar{r},\hat{\Omega},t)]^n d\bar{r}\,d\hat{\Omega}\,dt}\,, \quad (120$$

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.

APPENDIX B

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.

$$\omega^*(x) = \sum_{i=1}^{n} \omega_i\ \delta(x - x_i) \quad ,$$

BLANK PAGE

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

$$|C_1| = \begin{vmatrix} M_0 M_1 \\ M_1 M_2 \end{vmatrix}, \; |C_2| = \begin{vmatrix} M_0 M_1 M_2 \\ M_1 M_2 M_3 \\ M_2 M_3 M_4 \end{vmatrix}, \; \ldots, \; |C_{n-1}| = \begin{vmatrix} M_0 M_1 M_2 & \cdots & M_{n-1} \\ M_1 M_2 & \cdots & M_n \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ M_{n-1} M_n & --- & M_{2n-2} \end{vmatrix}$$

and

I_b) The roots of $Q_n(x)$ lie inside the interval $[a,b]$, i.e., $a \leq x_i \leq b$ whenever $Q_n(x_i) = 0$.

<u>Equivalence of Moments and Legendre Coefficients</u>

We shall use the following form for the Legendre expansion of an angular distribution:

$$f(\mu) = \sum_{\ell=0}^{\infty} \frac{2\ell + 1}{2} f_\ell \, P_\ell(\mu) \; . \qquad (1)$$

From this it follows that

$$f_\ell = \int_{-1}^{1} f(\mu) \, P_\ell(\mu) \, d\mu \;\; \text{and} \;\; f_0 \equiv 1 \; . \qquad (2)$$

The moments of the distribution are defined by

$$M_n = \int_{-1}^{1} \mu^n \, f(\mu) \, d\mu \quad . \qquad (3)$$

If the Legendre polynomials are written

$$P_\ell(\mu) = \sum_{n=0}^{\ell} p_{\ell n}\, \mu^n \quad , \qquad (4)$$

[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

$$f_\ell = \sum_{n=0}^{\ell} p_{\ell n} \int_{-1}^{1} f(\mu)\, \mu^n\, d\mu = \sum_{n=0}^{\ell} p_{\ell n} M_n \; . \qquad (5)$$

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

$$M_n = \sum_{\ell=0}^{n} \frac{2\ell + 1}{2} f_\ell\, p_{n\ell}^{-1} \qquad (6)$$

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

$$Q_{i+1}(x) = x^{i+1} + R_i(x) \ , \qquad (10)$$

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

$$Q_{i+1}(x) = x\ Q_i(x) + \sum_{k=0}^{i} d_{ik}\ Q_k(x) \ . \qquad (11)$$

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 tc 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

$$\begin{aligned} L_{i+1} &= E[Q_i(x)\ x^{i+1}] \\ &= \sum_{k=0}^{i} a_{i,k}\ M_{k+i+1} \ . \end{aligned} \tag{15}$$

Then the final orthogonality relation used in defining $Q_{i+1}(x)$ gives us

$$\begin{aligned} E[Q_{i+1}(x)\ Q_i(x)] = 0 \\ &= E[Q_{i+1}(x)\ x^i] + E[Q_{i+1}(x)\ R_{i-1}(x)] \\ &= E[x\ Q_i(x)\ x^i] - \mu_{i+1}\ E[Q_i(x)\ x^i] - \sigma_i^2\ E[Q_{i-1}(x)\ x^i] \\ &= L_{i+1} - \mu_{i+1}\ N_i - \sigma_i^2\ L_i \end{aligned}$$

or

$$\begin{aligned} \mu_{i+1} &= \frac{L_{i+1}}{N_i} - \sigma_i^2 \frac{L_i}{N_i} \\ &= \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}} \ . \end{aligned} \tag{16}$$

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

$$a_{i+1,k} = a_{i,k-1} - \mu_{i+1}\ a_{i,k} - \sigma_i^2\ a_{i-1,k} \ . \tag{17}$$

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 $M_o, M_1, \ldots, M_{2n-1}$ 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:

$$\begin{vmatrix} M_0 M_1 M_2 \cdots\cdots M_i \\ M_1 M_2 M_3 \cdots\cdots M_{i+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ M_i \; M_{i+1} \cdots M_{2i} \end{vmatrix} > 0 \; .$$

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 $Q_{n-i}(x)$; 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 $x_1 > x_2 > \ldots\ldots > x_{n-1}$ 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

$$\theta(x) = (x - x_1)(x - x_2)(x - x_3) \;....\; (x - x_s) \;,$$

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, ..., 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, ..., 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.

$$\begin{aligned} E[f(x)] &= E[q_{n-1}(x)\, Q_n(x)] + E[r_{n-1}(x)] \\ &= E[r_{n-1}(x)] \text{ from the orthogonality property of } Q_n . \end{aligned} \tag{18}$$

However, we want

$$E[f(x)] = \sum_{i=1}^{n} f(x_i)\cdot\omega_i = \sum_{i=1}^{n} q_{n-1}(x_i)\ Q_n(x_i)\cdot\omega_i + \sum_{i=1}^{n} r_{n-1}(x_i)\cdot\omega_i$$

$$= \sum_{i=1}^{n} q_{n-1}(x_i)\ Q_n(x_i)\cdot\omega_i + E[r_{n-1}(x)] \ . \qquad (19)$$

By subtracting Eq. (18) from Eq. (19), we find that we must require, for all polynomials, $q_{n-1}(x)$, that

$$\sum_{i=1}^{n} q_{n-1}(x_i)\ Q_n(x_i)\cdot\omega_i = 0 \ . \qquad (20)$$

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

$$E[Q_k(x)] = \sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i \quad \text{for } k = 0,1,\ldots,n-1 \ . \qquad (22)$$

However,

$$E[Q_k(x)] = E[Q_k(x)\ Q_o(x)] = N_o\ \delta_{ko} \ .$$

Thus we must have

$$\sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i = N_o\ \delta_{ko} \quad \text{for } k = 0,1,\ldots,n-1 \ . \qquad (23)$$

Multiplying Eq. (23) by $[Q_k(x_j)/N_j]$ and summing over k, we find

$$\sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} \sum_{i=1}^{n} Q_k(x_i)\cdot\omega_i = \sum_{i=1}^{n} \omega_i \{ \sum_{k=0}^{n-1} \frac{Q_k(x_j)\ Q_k(x_i)}{N_k} \}$$

$$= \sum_{k=0}^{n-1} \frac{Q_k(x_j)}{N_j} N_o\ \delta_{ko} = \frac{Q_o(x_j)}{N_o} N_o = 1\ . \tag{24}$$

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:

$$\frac{Q_n(x)\ Q_{n-1}(y) - Q_{n-1}(x)\ Q_n(y)}{N_{n-1}(x - y)}$$

$$= \frac{[(x - \mu_n)Q_{n-1}(x) - \sigma_{n-1}^2\ Q_{n-2}(x)]Q_{n-1}(y) - Q_{n-1}(x)[(y - \mu_n)Q_{n-1}(y) - \sigma_{n-1}^2 Q_{n-2}(y)]}{N_{n-1}(x - y)}$$

$$= \frac{(x - y)\ Q_{n-1}(x)\ Q_{n-1}(y) + \sigma_{n-1}^2\ [Q_{n-1}(x)\ Q_{n-2}(y) - Q_{n-2}(x)\ Q_{n-1}(y)]}{N_{n-1}(x - y)}$$

$$= \frac{Q_{n-1}(x)\ Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-1}(x)\ Q_{n-2}(y) - Q_{n-2}(x)\ Q_{n-1}(y)}{N_{n-1}(x - y)} \cdot \frac{N_{n-1}}{N_{n-2}}$$

$$= \frac{Q_{n-1}(x)\ Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-1}(x)\ Q_{n-2}(y) - Q_{n-2}(x)\ Q_{n-1}(y)}{N_{n-2}(x - y)}$$

$$= \frac{Q_{n-1}(x)\ Q_{n-1}(y)}{N_{n-1}} + \frac{Q_{n-2}(x)\ Q_{n-2}(y)}{N_{n-2}} + \frac{Q_{n-2}(x)\ Q_{n-3}(y) - Q_{n-3}(x)\ Q_{n-2}(y)}{N_{n-3}(x - y)}$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{Q_1(x)\, Q_0(y) - Q_0(x)\, Q_1(y)}{N_0(x - y)}$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{(x - \mu_1) - (y - \mu_1)}{N_0(x - y)} \tag{26}$$

$$= \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{1}{N_0} = \sum_{k=1}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} + \frac{Q_0(x)\, Q_0(y)}{N_0}$$

$$= \sum_{k=0}^{n-1} \frac{Q_k(x)\, Q_k(y)}{N_k} = D_{n-1}(x,y) \quad .$$

Therefore

$$D_{n-1}(x_j,x_i) = \frac{Q_n(x_j)\, Q_{n-1}(x_i) - Q_{n-1}(x_j)\, Q_n(x_i)}{N_{n-1}(x_j - x_i)} \tag{27}$$

For $i \neq j$ and $Q_n(x_j) = Q_n(x_i) = 0$,

$$D_{n-1}(x_j,x_i) = 0 \ .$$

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

$$\omega_j = \{D_{n-1}(x_j,x_j)]^{-1} = \left\{\sum_{k=0}^{n-1} \frac{Q_k^2(x_j)}{N_k}\right\}^{-1} \tag{28}$$

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

$$\Delta x_0 = \frac{Q_{i-1}(x_0)}{Q_i'(x_0)}\, \Delta\mu \quad . \qquad (29)$$

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

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

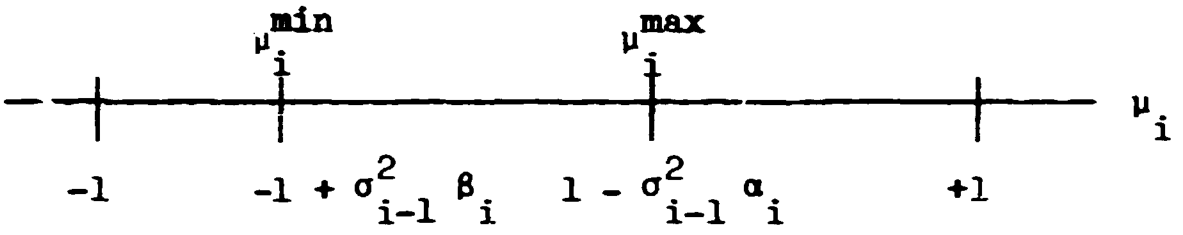
Note that

$$\alpha_i = \frac{Q_{i-2}(1)}{Q_{i-1}(1)} > 0 \ ,$$

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

$$1 - (\sigma_i^2)_{max} \frac{Q_{i-1}(+1)}{Q_i(+1)} = -1 - (\sigma_i^2)_{max} \frac{Q_{i-1}(-1)}{Q_i(-1)}$$

$$2 = (\sigma_i^2)_{max} \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

$$(\sigma_i^2)_{max} = 2 \Bigg/ \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

We can work back from the limits on μ_i and σ_i^2 to obtain limits on the moments.

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

$$N_i = \sum_{k=0}^{i} a_{ik} M_{k+i} = M_{2i} + \sum_{k=0}^{i-1} a_{ik} M_{k+i} \quad \text{since } a_{ii} = 1 \text{ .}$$

Therefore

$$0 < \sigma_i^2 < 2 \Bigg/ \left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)} \right]$$

implies

$$-\sum_{k=0}^{i-1} a_{ik} M_{k+i} < M_{2i} < \frac{2N_{i-1}}{\left[\frac{Q_{i-1}(+1)}{Q_i(+1)} - \frac{Q_{i-1}(-1)}{Q_i(-1)}\right]} - \sum_{k=0}^{i-1} a_{ik} M_{k+i}$$

$$\mu_{i+1} = \frac{L_{i+1}}{N_i} - \frac{L_i}{N_{i-1}}$$

$$L_{i+1} = \sum_{k=0}^{i} a_{ik} M_{k+i+1} = M_{2i+1} + \sum_{k=0}^{i-1} a_{ik} M_{k+i+1}$$

$$\mu_{i+1}^{max} = 1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)} \text{ ; therefore,}$$

$$L_{i+1}^{max} = N_i - N_i \sigma_i^2 \cdot \frac{Q_{i-1}(1)}{Q_i(1)} + \frac{N_i L_i}{N_{i-1}} = N_i \left(1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)}\right) + L_i \sigma_i^2$$

$$M_{2i+1} < N_i \left(1 - \sigma_i^2 \frac{Q_{i-1}(1)}{Q_i(1)}\right) + L_i \sigma_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k+i+1}$$

also

$$M_{2i+1} > N_i \left(- 1 - \sigma_i^2 \frac{Q_{i-1}(-1)}{Q_i(-1)}\right) + L_i \sigma_i^2 - \sum_{k=0}^{i-1} a_{ik} M_{k+i+1} \quad .$$

To obtain the limits on the Legendre coefficients, take the set of moments already determined M_1, M_2, ..., 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, ..., M_{2i-1} are combined with M_{2i}^{min} and converted, one obtains f_{2i}^{min}.

APPENDIX C

MORSE Input Instructions

There are five subroutines that read information for a complete MØRSE run. A description of the formats and variable definitions is given in this appendix.

The input read by Subroutine INPUT 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 (15I5)

NSTRT - number of particles per batch,

NMØST - maximum number of particles allowed for in the bank(s),

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 there are cross sections. Should be the 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 there are cross sections. Should be the 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,

MAXTIM - maximum clock time in minutes allowed for the problem to be on the computer (360/91 c.p.u. time),

MEDIA - number of cross-section media. Should agree with NMED read by subroutine XSEC,

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.

*See Table C-II, page C-8, for sample input.

BLANK PAGE

CARD C (4I5, 5E10.5)

ISØUR - source energy group if > 0,
if ISØUR ≤ 0, SØRIN is called for input,

NGPFS - number of groups for which the source spectrum is to be defined,

ISBIAS - no source energy biasing if set equal to zero; otherwise the source energy is to be biased,

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,

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

CARD D

$$\left.\begin{array}{l}\text{XSTRT}\\ \text{YSTRT}\\ \text{ZSTRT}\end{array}\right\}\begin{array}{l}\text{coordinates for source particles}\\ \text{(values may be overridden by subroutine SØURCE)}\end{array}$$

AGSTRT - starting age for source particles,

$$\left.\begin{array}{l}\text{UINP}\\ \text{VINP}\\ \text{WINP}\end{array}\right\}\begin{array}{l}\text{source particle direction cosines}\\ \text{if all are zero, isotropic directions are chosen}\end{array}$$

If ISØUR on card C is ≤ 0, subroutine SØRIN will be called for the input of source data. For the sample problem an input spectrum with biasing parameters is input.

CARDS E1(7E10.4)

NGPFS values of FS(I), the fraction of source particles in group I, are required.

CARDS E2(7E10.4)

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

Reminder: Cards E1 and E2 are not needed if ISØUR > 0.

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.

If a collision tape is desired (NCØLTP > 0) on card B, include Card G; otherwise omit.

CARD G (2I5, 5X, 36I1, 13I1)

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 (see Table C-I for definition of parameters).

NCØLLS(J), J=1, 13 - an index to indicate the types of collisions to be put on tape (BANKR arguments 1-13, page 166 for definition).

CARD H (Z12)

RANDØM - starting random number.

CARD I (14I5)

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 allowed if > 0,

NØLEAK - index indicating that non-leakage is allowed if > 0,

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

MXREG - maximum number of regions for which there are weight standards and exponential transform variables (will be set to one if ≤ 0),

MAXGP - maximum number of groups for which there are weight standards and exponential transform variables (will be set to one if ≤ 0).

If (NSPLT + NKILL + NPAST) = 0, omit cards J.

CARD J (6I5, 4E10.5) (see p. 41 of ref. 6)

NGP1
NDG
NGP2
NRG1
NDRG
NRG2

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).

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 $0 \leq \text{PATH} < 1$).

The above information is repeated until data for all groups and regions are input. If either NGP1 or NRG1 equal zero, the values will be stored for all MAXGP and MXREG.

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

The following cards are omitted if IEBIAS ≤ 0

CARDS K (7E10.4)

((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 (14I5)

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.

If MFISTP ≤ 0, omit cards M and N

CARDS M (7E10.4)

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

CARDS N (7E10.4)

(FSE(IG,IMED), IG=1, NMGP), 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.

For a combined problem, the following cards must be included; omit for a pure neutron or gamma-ray problem.

CARDS O (7E10.5)

((GWLØ (IG,NREG) IG = 1, NGPQTN or NGPQTG), NREG = 1, MYREG) - values of the weight to be assigned to the secondary particles being generated. NGPQTN groups are read for each region in a forward problem and NGPQTG for an adjoint. Input for each region must start on a new card.

Geometry input data

Read by subroutine JØMIN and the specific input depends on the geometry packages used (see ref. 1, pp. 49-53 and p. 18C, and see Appendix D).

XSEC input data

Read by subroutine XSEC.

CARD XA (20A4)

Title card for cross sections.

CARD XB (16I5)

NGP* - the number of primary groups for which there are cross sections to be stored. Should be same as NMGP on card B,

NDS - number of downscatters for NGP (usually NGP),

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

NDSG - number of downscatters for NGG (usually NGG),

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

INDS - number of downscatters for the INGP groups (usually INGP),

NMED - number of media for which cross sections are to be stored - should be same as MEDIA on card B,

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, including P_0,

NSCT - number of discrete angles (usually NCØEF/2 $)_{\text{Integral}}$,

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

IXTAPE - logical tape unit of binary cross-section tape, set = 0 if cross sections are from cards.

CARD XC (16I5)

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,

*See Table C-II, page C-8, for sample input.

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.

CARD XD (16I5)

Element identifiers for cross-section tape, omit if IXTAPE ≤ 0. Element identifiers must be in same order as elements are on tape.

The following cards are read by subroutine READSG.

CARD XE

ANISN format if IDTF ≤ 0; otherwise DTF-IV format. Cross sections for INGP groups with INDS downscatters for NELEM elements each with NCØEF coefficients.

The mixing cards are read by subroutine JNPUT.

CARDS XF (2I5, E10.5)

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).

RHØ - density of element KE in medium KM.

Analysis input data

Read by subroutine SCØRIN.

For the sample problem, the following cards are required:

CARD SA (20A4)

analysis title card

CARD SB (I10)

ND - number of detectors.

CARD SC (7E10.4)

RAD(I) I = 1, ND - detector radii.

CARD SD (7E10.4)

FDCF(I) I = 1, NGPQTN - response function for neutrons groups 1 through NGPQTN, see Card B.

Table C-I. Variables That May Be Written on Tape

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 I, page 8, and Table II, page 12.

Table C-II. 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)

	Problem Type					
Input Variable	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	
NGPQTN	14	0	14	0	14	CARD B Variables
NGPQTG	0	17	0	17	17	
NMGP	22	18	22	18	22	
NMTG	22	18	22	18	40	
NGP	22	18	22	0	22	CARD XB Variables
NGG	0	0	0	18	18	
INGP	22	18	40	40	40	

*For cross sections with full downscatter NDS = NGP, NDSG = NGG and INDS = INGP.

APPENDIX D

Geometry Input Instructions

SPHERICAL GEØM*

CARD GA (I5, D10.5)

MED - medium number interior to R (>0)

R - outer radius of sphere or spherical shell containing MED.

Repeat CARD GA for all radii ($\leq$20) in increasing order.

End CARD GA input with blank card.

CARD GB (D10.5)

R - region radius of sphere or spherical shell containing regions. Region numbers are assigned in consecutive order starting with 1, and R must be in increasing order.

Repeat CARD GB for the number of regions ($\leq$20).

End CARD GB input with blank card. If no regions are desired, a blank card must be used to signal no region geometry.

*Taken from ref. 6.

SLAB GEØM

CARD GA (I5, D10.5)

MED - medium with Z as lower bound (>0)

Z - lower limit of medium MED.

Repeat CARD GA for all boundaries with the last card containing MED = 0 and the boundary of the system.

CARD GB (D10.5)

Z - lower limit of region boundary. Region numbers are assigned in consecutive order starting with 1 and Z must be in increasing order.

Repeat CARD GB for all region boundaries.

End CARD GB input with a blank card. If no region geometry is desired a blank card is required.

CARD GC (4D10.5)

XL - lower boundary of system in X direction.

XU - upper boundary of system in X direction.

YL - lower boundary of system in Y direction.

YU - upper boundary of system in Y direction.

CYLINDRICAL GEØM

CARD GA (I5, 5X, A8)

NREGIN - flag to indicate material media (=1) or both region and material media (=2).

SEX - sex of programmer.

CARD GB (E10.5)

R - radii of the cylindrical shells describing the material media in ascending order.

Repeat CARD GB until all radii have been input.

End CARD GB input with a blank card.

CARD GC (E10.5, 12I5/8I5)

H - upper height of medium M(I) (>0).

Cylinders assumed to start at H = 0.

M(I) - media for the cylindrical shells for this height.

Repeat CARD GC until all height intervals have been input.

End CARD GC input with a blank card or if there are more than 12 radial intervals, 2 blank cards.

CARD GD (E10.5) omit if NREGIN = 1

RG - radii of the cylindrical shells describing the region geometry in ascending order.

Repeat CARD GD until all region geometry has been input.

End CARD GD with a blank card.

CARD GE (E10.5, 12I5/8I5) omit if NREGIN = 1

HG - upper height of region MG(I).

MG(I) - region numbers for the cylindrical shells for this height.

Repeat CARD GE until all height intervals have been input.

End CARD GE input with a blank card or if there are more than 12 radial intervals, 2 blank cards.

GENERAL GEØM

CARD GA (I5, 5X, A6, 1X, A7) Hollerith left adjusted

NSTAT - flag to indicate material media only if 1 and both region and material media if 2.

SEX - sex of the programmer. (Select one from MALE, FEMALE,or blank indicating uncertain.)

STATUS - marital status of programmer.

CARD GB (2A4, A3, 5(D10.5, A1)

DUMMY(3) - hollerith characters not used.

FIN(I) - zone boundaries in increasing order along the X-axis.

BCD(I) - flag to indicate end of input if blank, comma means to continue.

Repeat CARD GB if more than five boundaries along the Y axis are needed.

CARD GC - same as CARD GB except for Y axis.

Repeat CARD GC if more than five boundaries along the Y axis are needed

CARD GD - same as CARD GB except for Z axis.

Repeat CARD GD if more than five boundaries along the Z axis are needed.

CARD GE (A4, A2, 3I5)

BCD1 - hollerith ZONE

BCD2 - dummy

NXZNO - integers which specify the zone as being the NXZNOth

NYXNO - zone in the X direction, NYZNOth zone in the Y

NZZNO - direction, and NZZNOth zone in the Z direction.

CARD GF(2A4, A3, 5(D10.5, A1)

DUMMY(3) - hollerith characters not used.

FIN(I) - block boundaries in increasing order along the X axis.

BCD(I) - flag to indicate end of input if blank, comma means to continue.

Repeat CARD GF if more than five boundaries along the X axis are needed.

CARD GG - same as CARD GF except for Y axis.

Repeat CARD GG if more than five boundaries along the Y axis are needed.

CARD GH - same as CARD GF except for Z axis.

Repeat CARD GH if more than five boundaries along the Z axis are needed.

CARDS GI to GO describe the geometry for a block and must be included for each block in the zone.

CARD GI (A4, A2, 3I5)

BCD1 - hollerith BLOC

BCD2

NXBND - integers which specify the block as being the

NYBND - NXBNDth in the X direction, the NYBNDth in the

NZBND - Y direction, and the NZBND in the Z direction.

CARD GJ (3A4, 10(I5,A1)

NAM2 - hollerith MEDI

DUM(2) - dummy

INP(I) - a list of media sector by sector in the block

BCD(I) - flag to indicate end of input if blank, a comma means to continue.

Continuation with 12(I5,A1) format is permissible.

CARD GK (3A4, 10(I5,A1))

NAM2 - hollerith SURF

DUM(2) - dummy

INP(I) - a list of quadratic surfaces appearing in the block. Numbers must appear in the order the surfaces are described on CARD GQ.

BCD(I) - flag to indicate end of input of blank, a comma means to continue.

Continuation of CARD GK in 12(I5,A1) format is permissible.

CARD GL (A4, A2, 18I3)

S1 - hollerith SECT

DUM - dummy

IND(I) - the designation of each sector which describes the position of the sector relative to quadratic surfaces.

+1: sector is on positive side of surface,

-1: sector is on negative side of surface,

0: surface is not needed to define sector.

There must be a CARD GL for each sector and references to quadratic surfaces must be in same order as they are listed on CARD GQ.

CARD GM (3A4, 10(I5,A1)

NAM2 - hollerith REGI

DUM(2) - dummy

INP(I) - a list of regions sector by sector in the block.

BCD(I) - flag to indicate end of input if blank, a comma means to continue.

Continuation with 12(I5,A1) format is permissible.

CARD GN (3A4, 10(I5,A1))

NAM2 - hollerith SURF

DUM(2) - dummy

INP(I)
BCD(I) } same as for CARD GK except for region input instead of material input.

CARD GO(A4, A2, 18I3)

S1 - hollerith SECT

DUM - dummy

IND(I) - same as for CARD GL except for region input instead of material input.

<u>Repeat CARDS GI to GØ for each block.</u>

CARD GP (I5, 16A4, A2)

NØBD - total number of quadratic surfaces in the entire system.

DUM(I) - hollerith characters ignored by the code. (Helpful in identifying input at a later time.)

CARD GQ (4(D10.5, A4, 1X, A1))

CØF(J) - coefficient of the term

BCD1(J) - hollerith indicating which term of the equation. XSQ, YSQ, ZSQ, XZ, YX, YZ, XY, ZX, YZ, X, Y, Z, or blank are the possibilities.

BCD2(J) - a flag which indicates the quadratic equation continues. Any non-blank character ends the field. The next function must start on new card.

Repeat CARDS GQ until all surfaces have been described.

A sample of the input is shown in Figure D.1.

CODED BY

DATE REQUEST NO.

FORTRAN

PROBLEM

PROGRAM

PAGE 1 OF 2

STATEMENT NUMBER | TYPE | CONT | FORTRAN STATEMENT | IDENTIFICATION

```
    2      MALE                                                         CARD GA
X ZONE          -30.5,      30.5                                        CARD GB
Y ZONE          -30.5,      30.5                                        CARD GC
Z ZONE            0.0,     238.                                         CARD GD
ZONE      1    1    1                                                   CARD GE
X BLOCK         -30.5,      30.5                                        CARD GF
Y BLOCK         -30.5,      30.5                                        CARD GG
Z BLOCK           0.0,      14. ,      62.8,    182.5,     238.0        CARD GH
BLOCK     1    1    1                                                   CARD GI
MEDIA        1000,    1                                                 CARD GJ
SURFACES        1                                                       CARD GK
SECTOR -1                                                               CARD GL
SECTOR +1                                                               CARD GL
BLOCK     1    1    2                                                   CARD GI
MEDIA        1000,    1                                                 CARD GJ
SURFACES       2                                                        CARD GK
SECTOR -1                                                               CARD GL
SECTOR +1                                                               CARD GL
BLOCK     1    1    3                                                   CARD GI
MEDIA        1000,    3     1                                           CARD GJ
```

CODED BY ______ DATE ______ REQUEST NO. ______

FORTRAN

PROBLEM ______
PROGRAM ______
PAGE 2 OF 2

```
SURFACES        3,    4                                                 CARD GK
SECTOR -1  0                                                            CARD GL
SECTOR +1 -1                                                            CARD GL
SECTOR  0 +1                                                            CARD GL
BLOCK     1    1    4                                                   CARD GI
MEDIA        1000.    1                                                 CARD GJ
SURFACES        5                                                       CARD GK
SECTOR -1                                                               CARD GL
SECTOR +1                                                               CARD GL
    5EQUATIONS FOR DOUBLE TAPERED COLLIMATOR                            CARD GP
      1.  XSQ       1.    YSQ   -161.29        $                        CARD GQ
      1.  XSQ       1.    YSQ   -.0225772 ZSQ  +4.42615 Z               CARD GQ
-218.87         $                                                       CARD GQ
      1.  XSQ       1.    YSQ   -29.160        $                        CARD GQ
      1.  XSQ       1.    YSQ   -40.3225       $                        CARD GQ
      1.  XSQ       1.    YSQ   -.060933 ZSQ   +19.5747 Z               CARD GQ
-1572.0868      $
```

APPENDIX E

Library Subroutines and Functions

The following subroutines and functions are library routines at Oak Ridge National Laboratory and are not provided with MORSE.

Subroutine or Function	Called From	Purpose
LØC	Main, XSCHLP, HELP	Determine absolute address of cell given as argument.
INTØBC	DATE	Converts integer to EBCDIC.
INTBCD	DATE, TIMER, SUBRT	Converts integer to EBCDIC and returns number of bytes in EBCDIC string.
FETYPE	READSG	Determines if a character is a number or a letter.
BCDTØI	READSG	Converts EBCDIC to integer.
ICØMPA	INPUT, BNKHLP, HELPER	Compares bit by bit N bytes of two variables; returns zero if the two variables are identical.
MØDEL	INPUT	Determines whether the problem is being executed on the IBM-360 model 75 or 91.
IDAY	IWEEK	Determines number of the month, day, and year.
ICLØCK	TIMER, MØRSE	Determines c.p.u. time.
INSERT	TIMER	Inserts a string of given length at a specified point in another string of characters.

There are several uses of these library routines. 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 XSHLP, BNKHLP, and HELP.

Several other uses of these routines are made, but they are relatively unimportant. MODEL is used to scale MAXTIM, depending on the machine on which the job is being executed. ICØMPA is used by INPUT to terminate a job when a non-blank or alphanumeric character appears in the first column of Card A. READSG has an option of checking for sequence errors in the cross-section cards. While none of these features are necessary to the operation of MORSE, they have proven to be quite useful.