

PARTISN – Introduction and System Overview

Transport Methods Group, CCS-4
Los Alamos National Laboratory

1

CCS-4 – Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	1-3
LIST OF FIGURES	1-5
INTRODUCTION	1-7
DOCUMENTATION SUMMARY.....	1-9
COMMON MODELING CONCEPTS	1-11
Geometry Concepts	1-11
Iteration Strategy	1-12
Modular Structure and Interface Files.....	1-15
REFERENCES	1-17

LIST OF FIGURES

Figure 1.1: Spatial mesh labeling convention in PARTISN.....	1-11
Figure 1.2: Simplified flow diagram of SOLVER iteration strategy.....	1-14
Figure 1.3: PARTISN Structure.....	1-15



INTRODUCTION

This document is the central user methods and programming documentation for PARTISN. PARTISN is an acronym for PARallel, TIme-dependent SN.

The PARTISN code package is a modular computer program package designed to solve the time-independent or dependent multigroup discrete ordinates form of the Boltzmann transport equation in several different geometries. The modular construction of the package separates the input processing, the transport equation solving, and the post processing (or edit) functions into distinct code modules: the Input Module, the Solver Module, and the Edit Module, respectively. PARTISN is the evolutionary successor to the DANTSYSTM code system package. The Input and Edit Modules in PARTISN are very similar to those in DANTSYS. However, unlike DANTSYS, the Solver Module in PARTISN contains one, two, and three-dimensional solvers in a single module. In addition to the diamond-differencing method, the Solver Module also has Adaptive Weighted Diamond-Differencing (AWDD), Linear Discontinuous (LD), and Exponential Discontinuous (ED) spatial differencing methods. The spatial mesh may consist of either a standard orthogonal mesh or a block adaptive orthogonal mesh. The Solver Module may be run in parallel for two and three dimensional problems.

Some of the major features included in the PARTISN code package are:

- a free-field format ASCII text input capability;
- the use of a diffusion or transport synthetic acceleration scheme to accelerate the iterative process in the Solver Module;
- direct (forward) or adjoint calculational capability;
- standard plane, two-angle plane, cylindrical or spherical geometry options for 1-d;
- x-y, r-z and r-theta geometries in 2-d;
- x-y-z and r-z-theta geometries in 3-d;
- arbitrary anisotropic scattering order;
- vacuum, reflective, periodic, white, or surface source boundary condition options;
- inhomogeneous (fixed) source or k_{eff} calculation options as well as time-absorption (alpha), nuclide concentration, or dimensional search options for time-independent calculations;
- time-dependent calculations with time-dependent sources and/or cross sections;

- a variety of spatial differencing methods;
- a ray trace first collision option to obtain a first collision source from an arbitrary source distribution;
- a strictly positive scattering source option;
- an automatic mesh coarsening option;
- user flexibility in using both ASCII text or sequential file input;
- user flexibility in controlling the execution of both modules and submodules;
- extensive, user-oriented error diagnostics.

PARTISN is a large, very flexible code package. Great effort has been devoted to making the code highly user-oriented. Simple problems can be easily run and many of the code options can be ignored by the casual user. At the same time numerous options for selective and sophisticated executions are available to the more advanced user. In all cases, redundancy of input has been minimized, and reasonable default values for many input parameters are provided. The input is designed to be meaningful, easily understood, easily verified, easy to change, and logically common. The printed output is well documented with liberal use of descriptive comments and headings.



DOCUMENTATION SUMMARY

The S_n Code package includes documentation for varying audiences.

Table 1.1 Documentation Elements

Documentation Type	Title
Methods	Methods Document
Introduction	Introduction and System Overview
User's Guide	PARTISN User's Guide
Details	Details of the Geometry and Solver Input
Details	Running the Edit Module
Reference	Free Field Input Reference
Reference	Cross Section Libraries
Details	Material Mixing Tutorial
Methods	Methods Document
Details	Code Structure
Reference	Error Messages
Reference	File Descriptions
Reference	Solver Module Abstracts
Reference	Bibliography

As in the case of PARTISN, this manual has evolved from the DANTSYS manual (Ref. 1). It contains all of the DANTSYS documentation (excepting the User's Guides), but with expanded sections where necessary to cover the new capabilities in PARTISN.

COMMON MODELING CONCEPTS

Geometry Concepts

In the specification of geometry and space-variable related input, the user must be familiar with the nomenclature used by PARTISN. The terms fine mesh, coarse mesh, and zones are defined below for the orthogonal geometry mesh used in PARTISN.

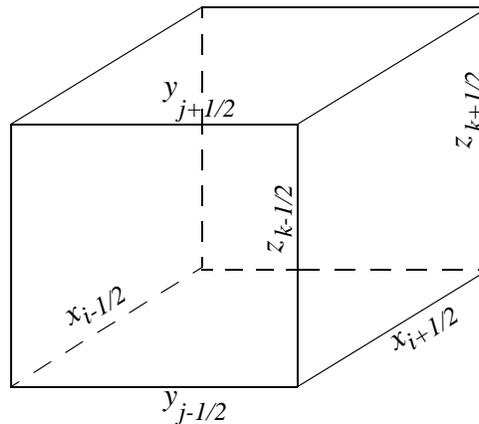


Figure 1.1 Spatial mesh labeling convention in PARTISN.

The fine mesh is the spatial solution-mesh for the problem, as depicted in Figure 1.1 above. Each fine mesh, or fine mesh interval, is bounded by an adjacent pair of fine-mesh grid-surfaces $x_{i-1/2}$ and $x_{i+1/2}$ with $x_{i-1/2} < x_{i+1/2}$ in the x direction; $y_{j-1/2}$ and $y_{j+1/2}$ with $y_{j-1/2} < y_{j+1/2}$ in the y direction; $z_{k-1/2}$ and $z_{k+1/2}$ with $z_{k-1/2} < z_{k+1/2}$ in the z direction. There are IT, JT and KT such fine mesh intervals respectively. No material discontinuities may occur within a fine mesh interval. The specification of the fine mesh is accomplished by specifying how many equally sized fine mesh intervals there are in each coarse mesh.

The coarse mesh is a spatial superset of the fine mesh and is formed by partitioning the spatial domain of the problem into a suitable number of “coarse” intervals. There are IM, JM, and KM coarse mesh intervals in each of the coordinate directions spanning the problem. Each coarse mesh interval contains one or more fine mesh intervals. All fine

mesh intervals within a coarse mesh interval have equal widths. No material discontinuities may occur within a coarse mesh interval.

The zone is a spatial superset of coarse mesh intervals and is characterized by a single set of multigroup nuclear properties, i.e., cross sections, so that all fine mesh intervals within a zone have the same cross sections. The user assigns a zone number to each coarse mesh interval. The zone number tells the code which macroscopic cross section set is to be used within that zone. Coarse mesh intervals having the same zone number need not be simply connected.

A zone number of 0 (zero) can be used to specify that a coarse mesh interval is a pure void (all cross sections are identically zero).

In the block adaptive mesh, the coarse meshes define the block structure. Then, within each individual block (coarse mesh), an arbitrary level of coarsening is applied to that block's fine mesh, resulting in a block adaptive mesh structure. This structure is used only within the Solver Module; the Input and Edit Modules see only the user-input coarse/fine (single level) mesh.

More detail on these concepts is to be found on page 3-15.

Iteration Strategy

In this section is described the basic iteration strategy used in the execution of the Solver Module. A more detailed description of the strategy is given on page 3-17 including the iteration controls that the user inputs and a discussion of the iteration monitor printout that is printed in the output.

The basic features of the iteration strategy are shown in the simplified flow diagram in Figure 1.2.

The iterative strategy is basically divided into three parts: inner iterations, outer iterations, and eigenvalue search iterations.

The inner iterations are concerned with the convergence of the pointwise scalar fluxes in each group due to iteration on within-group scattering processes. For eigenvalue problems, the source to each group is given by the fission source from the previous outer iteration plus any in-scattering sources. For fixed source problems, the source to each group is the input source distribution plus the in-scattering source.

The outer iterations are concerned with the convergence of the eigenvalue, the fission source distribution and the energy-group upscatter source if any or all are present.

The eigenvalue search iteration is the ability of the code to adjust some parameters of the problem, namely the isotopic concentrations or the spatial dimensions of selected coarse mesh intervals, to obtain a desired value of the k_{eff} . Also the alpha eigenvalue (time constant) of the system is determined by a search procedure based upon successive determinations of k_{eff} .

Both the inner and outer iterations are accelerated using the diffusion synthetic method. See page 8-14 for the theory of this method.

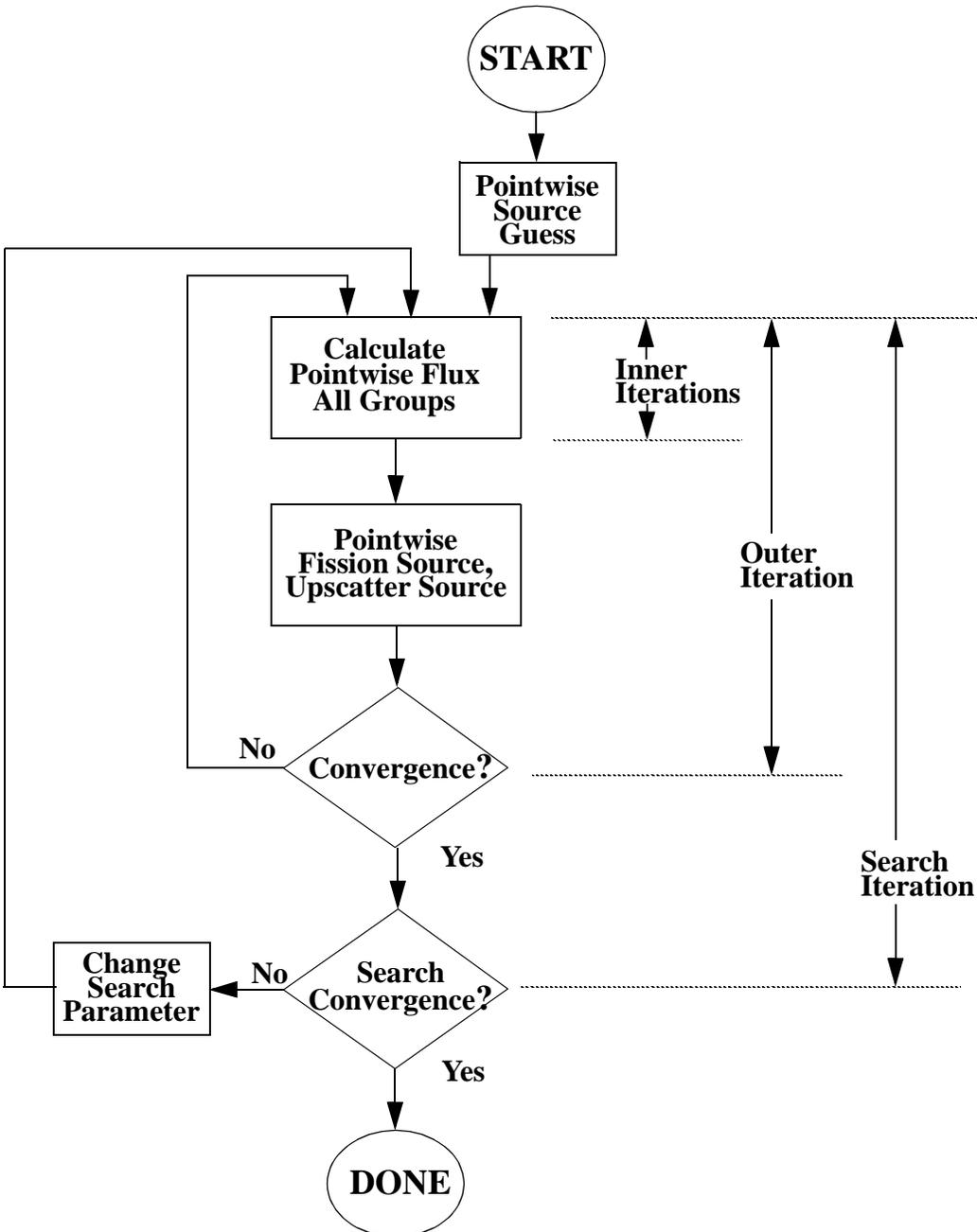


Figure 1.2 Simplified flow diagram of SOLVER iteration strategy

REFERENCES

1. R. E. Alcouffe, R. S. Baker, F. W. Brinkley, D. R. Marr, R. D. O'Dell, and W. F. Walters, "DANTSYS: A Diffusion Accelerated Neutral Particle Transport Code System," Los Alamos National Laboratory Manual LA-12969-M (June 1995)

PARTISN USER'S GUIDE

Transport Methods Group, CCS-4
Los Alamos National Laboratory

2

CCS-4 — Transport Methods Group

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

An Affirmative Action/Equal Opportunity Employer

PARTISN is a trademark of the Regents of the University of California, Los Alamos National Laboratory.

This work was supported by the US Department of Energy.

DISCLAIMER

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



USER'S GUIDE FOR PARTISN: A CODE PACKAGE FOR PARALLEL, TIME-DEPENDENT SN TRANSPORT

by
**Ray E. Alcouffe, Randal S. Baker, Jon A. Dahl,
and Scott A. Turner**

TABLE OF CONTENTS

TABLE OF CONTENTS.....	2-5
LIST OF FIGURES	2-7
LIST OF TABLES.....	2-9
INTRODUCTION	2-11
DOCUMENTATION FOR PARTISN USAGE.....	2-13
What Is In This User's Guide.....	2-13
What Is Available Elsewhere	2-14
PARTISN INPUT OVERVIEW	2-17
Input Block Order.....	2-17
Free Field Input Summary.....	2-19
Arrays	2-19
Numeric Data Items.....	2-19
Character Data Items	2-20
Blocks	2-20
Strings.....	2-20
Comments.....	2-20
Operators	2-20
Frequently Used Operators.....	2-22
MINI-MANUAL Introduction	2-23
MINI MANUAL	2-24
PARTISN INPUT DETAILS	2-33
Introduction	2-33
Title Line Details	2-36
Title Line Control.....	2-36
Block-I Details: Dimensions and Controls	2-37
Dimensions.....	2-37
Storage Requirements.....	2-39
Fine Mesh Mixing	2-39
Run Configuration Controls	2-40
Block-II Details: Geometry	2-41
Geometry Arrays	2-41
Block-III Details: Nuclear Data	2-42
Nuclear Data Type and Options	2-42
Alternate Library Name.....	2-45
Text Cross-Section Library Format.....	2-47
Block-IV Details: Cross-Section Mixing	2-49
Primary Mixing Arrays.....	2-50
MATLS input array	2-50
ASSIGN input array	2-51
PREMIX input array.....	2-51
Character Names vs. Numeric Names.....	2-52
Mixing Array for a Concentration Search.....	2-53
ASGMOD input array	2-53
Concentration Modifier	2-54
Miscellaneous Mixing Input.....	2-54

Block-V Details: Solver Input	2-56
Desired Calculation.....	2-56
Iteration Controls	2-57
Acceleration Controls	2-57
K-Code Convergence.....	2-59
Output Controls	2-59
Miscellaneous Solver Input	2-61
Quadrature Details	2-63
Transport Solver Details	2-65
Flux Guess From a File.....	2-67
General Eigenvalue Search Control.....	2-67
Dimension Search Input.....	2-68
Concentration Search Input	2-69
Parallelization Details	2-69
Mesh Potential Input.....	2-70
Time Dependent Input	2-71
Volumetric Source Options	2-72
Boundary Source Input	2-74
Boundary Source Vector Input Combinations.....	2-76
First Collision Source Input.....	2-77
Block-VI Details: Edit Input.....	2-79
Edit Spatial Specifications	2-79
Reaction Rates from Cross Sections.....	2-80
Edit Cross-Section Types by Position and Name	2-81
Reaction Rates from User Response Functions.....	2-82
Energy Group Collapse Specifications	2-83
Reaction Rate Summing	2-84
Mass Inventories	2-84
Power Normalization	2-85
Miscellaneous Edit Items.....	2-86
Special Plot Linkage	2-87
MENDF Library Edit Cross Sections	2-88
 REFERENCES	 2-89
 APPENDIX A: SAMPLE INPUT	 2-91
Sample Problem: Standard k_{eff} Calculation	2-91
Sample Problem: Output Listing	2-97
 APPENDIX B: OPERATING SYSTEM SPECIFICS	 2-107
UNIX/UNICOS Execution	2-107
Library Search Path	2-108
 PARTISN CHAPTER INDEX	 2-109

LIST OF FIGURES

Figure 2.1: PARTISN Input Order	2-18
Figure 2.2: Orientation of Faces	2-64

LIST OF TABLES

Table 2.1: LIBNAME Availability	2-42
Table 2.2: UNIX Search Path.....	2-104

INTRODUCTION

The PARTISN code package is a modular computer program designed to solve the multi-dimensional, time-independent or dependent, multigroup discrete-ordinates form of the Boltzmann transport equation.^{1,2}

PARTISN^{TM*} is based on the modular construction of the DANTSYS^{TM,3} code system package. This modular construction separates the input processing, the transport equation solving, and the postprocessing, or edit functions, into distinct, independently executable code modules, the INPUT, SOLVER, and EDIT modules, respectively. These modules are connected to one another solely by means of binary interface files. Standardized data and file management techniques as defined⁴ and developed by the Committee Computer Code Coordination (CCCC) are used for these files.

Some of the major features included in the PARTISN package are:

1. a free-field format text input capability, designed with the user in mind,
2. the use of diffusion synthetic acceleration⁵ or transport synthetic acceleration⁶ to accelerate the iterative process in the SOLVER module,
3. direct (forward) or adjoint calculational capability,
4. 1-D (slab, two-angle slab, cylindrical, and spherical), 2D (x-y, r-z, and r-theta) and 3-D (x-y-z and r-z-theta) geometry options,
5. arbitrary anisotropic scattering order via standard PN expansions or Galerkin scattering,
6. vacuum, reflective, periodic, white, or surface source boundary condition options,
7. inhomogeneous (fixed) source or k_{eff} calculation options, as well as time-absorption (α), nuclide concentration, or dimensional search options,
8. “diamond-differencing,” adaptive weighted diamond differencing (AWDD), linear discontinuous or exponential discontinuous spacial differencing schemes for the solution of the transport equation,
9. a diffusion solver that uses the multigrid⁷ or conjugate gradient method,

* - DANTSYS and PARTISN are trademarks of the Regents of the University of California, Los Alamos National Laboratory

10. user flexibility in using either ASCII text or sequential file input,
11. a ray trace first collision option to obtain a first collision source from an arbitrary source distribution (volume and external boundary sources),⁸
12. user flexibility in controlling the execution of both modules and submodules,
13. extensive, user-oriented error diagnostics,
14. a strictly positive scattering source⁹ option,
15. an automatic mesh coarsening¹⁰ option,
16. block Adaptive Mesh Refinement (AMR),¹¹
17. time-dependent calculations,¹² and
18. parallel processing.¹³⁻¹⁵

DOCUMENTATION FOR PARTISN USAGE

The documentation described here constitutes a complete manual for the use of the PARTISN code.

What Is In This User's Guide

This User's Guide is a chapter from the much larger PARTISN system document. This Guide provides the ASCII text input specifications for PARTISN.

The guide is intended to serve as a complete input manual for two classes of user. Special, succinct sections containing summaries and compact tables are intended for the advanced user in order to make his input preparation more efficient. The main body of the guide concerns itself with descriptions of the input and should be sufficient for the user familiar with discrete ordinates concepts. Novice users may find other chapters of the document necessary.

This Guide first gives an overview of the input block order required by the code.

Next is a "mini-manual" in which are listed all the names of available input arrays arranged by input block. Definitions of input arrays are not given, as the names are suggestive, but expected types and sizes are provided. This mini-manual is very useful to the user as a quick check for completeness, a quick reference to type and size, and as an index into the more detailed array descriptions that follow. For the experienced user, the mini-manual is frequently all that is needed to prepare a complete input deck.

Following the mini-manual are reference sections describing in detail all the input parameters and arrays.

Appendix A provides a sample PARTISN problem with explanation of the output for the user.

Lastly, Appendix B details operating system specifics, including how to effect an execution of the code.

Information of a reference, background, or theoretical nature that the first time user may need may not be found in this User's Guide, but the user will encounter liberal references to other chapters of this document for that sort of information.

What Is Available Elsewhere

In addition to this User's Guide, the user, especially the first time user, may find the information below described in other chapters of this document pertinent. For even greater detail on some of the general items, particularly the methods items, the user should look at Ref. 16.

The chapter "DETAILS OF THE BLOCK-I, GEOMETRY, AND SOLVER INPUT" starting on page 3-1 discusses in more detail the geometry and solver concepts and their related input. If the User's Guide proves insufficient for your needs, look in this chapter. Among the many sections of the chapter are ones on the input of inhomogeneous sources and a discussion of eigenvalue searches. There is also more detail on the Block-I input.

A discussion of how the EDIT module works and more detail on preparing the input is given in the chapter "RUNNING THE EDIT MODULE" starting on page 4-1.

The chapter "FREE FIELD INPUT REFERENCE" starting on page 5-1 serves as the reference manual for the free-field input (rules, format, and operators) used in this code. That chapter is summarized in this guide, but should the summary prove inadequate, the user is referred there for full details.

The chapter "CROSS-SECTION LIBRARIES" starting on page 6-1 gives details of the many library formats available to PARTISN, including sections on how to prepare your own card-image (or text) libraries.

The chapter "MATERIAL MIXING TUTORIAL" starting on page 7-1 describes the mixing concepts in detail and shows some examples.

Next is the chapter "PARTISN — METHODS MANUAL" starting on page 8-1. That chapter describes the theoretical basis for the PARTISN code.

In the chapter "PARTISN — CODE STRUCTURE" starting on page 9-1 is shown a brief overview of the code package. Included are sections on programming practices and standards, code package structure, and functional descriptions of the three principal modules comprising the package. In particular, the code package structure must be understood in order to make up input for piecewise executions of the code that are possible with controls that are part of the input in Block-I.

Error diagnostics that the user might encounter are found in the chapter "ERROR MESSAGES" starting on page 10-1. Several examples of input errors and the resulting error messages are provided for the user.

The chapter “FILE DESCRIPTIONS” starting on page 11-1 is a reference that describes all the files used by the package. Included is a detailed description of the file structure of the code dependent, binary, sequential interface files generated by and used in the PARTISN code package. Also included are descriptions of any other files produced or used by the package, both binary and text. In some cases, this may simply be a reference to a more comprehensive document, such as the file descriptions for the CCCC standard interface files.

PARTISN INPUT OVERVIEW

Input Block Order

The full PARTISN input consists of a title line section, followed by six blocks of free field input. The title line section is not free field. Any input referred to as a block uses the free field input form.

Block-I consists of basic control and dimensional information that allows efficient packing of the array data. This information also allows checking of the lengths of arrays supplied by interface files.

Block-II contains the geometric information.

Block-III consists of the nuclear data specifications.

Block-IV contains mixing information.

Block-V contains the rest of the input needed for specifying the flux calculation.

And lastly, Block-VI contains the edit (i.e., report writing) specifications.

If a text cross-section library is to be included in the input deck, it should be placed between Blocks III and IV. PARTISN supports many library formats and so the library may or may not be in free field format depending upon the option chosen.

A full input would then look like that diagrammed on the following page.

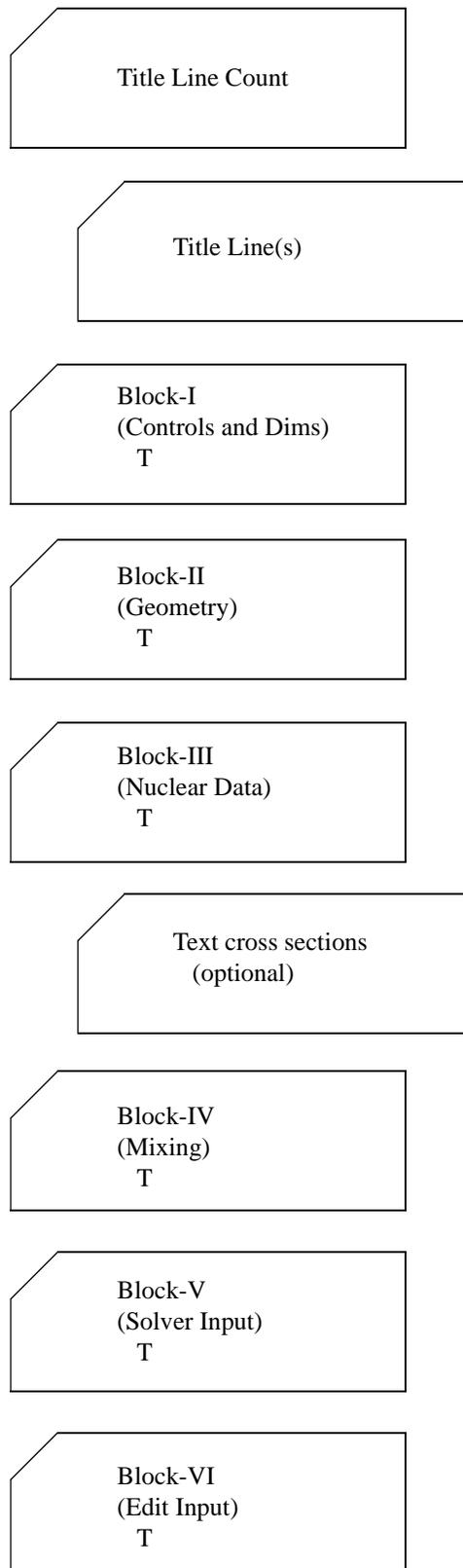


Figure 2.1 PARTISN Input Order

Free Field Input Summary

The chapter “FREE FIELD INPUT REFERENCE” starting on page 5-1 is summarized here for quick reference.

There are four basic input quantities in the free field input used in PARTISN; they are ARRAY, DATA ITEM, BLOCK, and STRING. Each of these is briefly described below along with the concept of an input operator.

Arrays

The “Array” is the most basic concept in the input. Data are given to the code by placing data items in an “Array.” To make an input to an array, one simply spells out the array name, appends an equal sign, and follows that with the data items to be entered into the array. For example, input for the x distribution of the volumetric source, for which the unique array name is SOURCX, might look like:

```
SOURCX= 0 0 0 1.1 1.1 0 0 0 0 0
```

The above input would enter source values of zero for the first three intervals, 1.1 for the next 2 intervals, and then fill the rest of the ten positions in the array with zero.

Data items within an array are separated by blanks or commas. In general, blanks may be used freely throughout except within a data item, within an array name, or between an array name and its equal sign.

Single value input variables are treated as arrays of unit length.

Numeric Data Items

Numeric data items follow a Fortran input convention. For example, all of the following are valid entries for the number ten:

```
10, 1.0+1, 1E1, 10.0
```

If a decimal point is not entered, it is assumed to be after the right-most digit.

Some arrays expect integer values for input. For such arrays, any input values containing a decimal point will be truncated.

Character Data Items

Character data items follow a Fortran variable name convention in that they are composed of up to eight characters, the first of which must be alphabetic with the rest alphanumeric. However, special characters and blanks may be included if the data item is surrounded by double quotes. Operators may NOT be used with character data items.

Blocks

Arrays are entered in groups called blocks. A block consists of one or more arrays (in any order) followed by the single character T. Thus T is the block delimiter.

Strings

Arrays may need to be entered in smaller pieces called strings. Strings are delimited with a semicolon(;). When there is matrix or other 2-d input, strings are frequently used to input information by row rather than for the whole 2-d array at once. The code dictates this, the user has no choice. The user is made aware of which arrays require string input through use of a certain notation, described later, in the input array descriptions.

Comments

A slash (/) may be used to enter comments in the input stream. After a slash is read no further processing of that card-image is done.

Operators

Several data operators are available to simplify the input.

The data operators are specified in the general form

$$n \ O \ d$$

where:

n is the “data numerator,” either an integer or a blank;
O is any one of the “data operator” characters shown below; and
d is a “data entry” (may be blank for some operators).

Note: The “data operator” character must be appended to the “data numerator.”

Using operators, the SOURCX input described above could more succinctly be given as:

SOURCX= 0 0 0 2R 1.1 F0

Note that the operators for FIDO-like repeat and fill were used and were appended directly to the data numerator. In general, all the FIDO¹⁷ operators may be used in numeric entry.

A table of the most used operators is given next including brief descriptions. For full descriptions of these and a complete list of all the available operators, including the more esoteric ones, the user is referred to “FREE FIELD INPUT DETAILS” on page 5-13.

Frequently Used Operators

Operator ^a	Functionality
nR d	REPEAT the data item d, n times.
nI d	INTERPOLATE (linear) n data items between data item d and the next data item.
nC d	SCALE (multiply) the n previous entries by d.
F d	FILL the rest of the data string with the data item d.
nY m	STRING REPEAT. Repeat the previous m strings, n times.
nL d	INTERPOLATE LOGARITHMICALLY n data items between d and the next d.
nZ	ZERO. Enter the value zero n successive times.
nS	SKIP. Skip the next n data items.
nQ m	SEQUENCE REPEAT. Enter the last m entries, n more times.
nG m	SEQUENCE REPEAT WITH SIGN CHANGE. Same as the Q option but the sign of the m entries is changed every repeat.
nN m	SEQUENCE REPEAT INVERT. Same as the Q option but the order of the m entries is inverted each repeat.
nM m	SEQUENCE REPEAT INVERT WITH SIGN CHANGE. Same as N option but the sign is also changed every repeat.
nX	COUNT CHECK. Causes code to check the number of entries in the current string so far, against the number n.

- a. The operator character must always be appended directly to n. d or m need not be immediately adjacent to the operator character.

MINI-MANUAL Introduction

On the following few pages is given a complete list of the input names, expected array sizes, and order within the array. No description of the array contents is given in this MINI-MANUAL as full details are given in later sections. The MINI-MANUAL is intended to serve as a quick reference for the knowledgeable user.

In both the MINI-MANUAL and in the detailed sections which follow, a shorthand form is used to indicate the size and order of the array that the code expects. This information is enclosed in square brackets immediately after the array name. Essential features are:

1. A single entry in the brackets is the array length.
2. No brackets at all indicates a simple variable (i.e., an array of unit length).
3. A dash (-) in the brackets indicates an arbitrary length.
4. A semicolon (;) indicates that the input for that array is expected in strings. To the left of the semicolon is the string length. To the right of the semicolon is the number of strings in the array.
5. If the number of strings is shown as a product, the order is important. The left-most quantity must be exhausted first, then, the next one to the right is varied. For example, the array name for the full spatial source distribution is shown as:

SOURCEF [IT;JT*KT*NMQ]

where - IT is the number of fine meshes in the X-direction, JT is the number of fine meshes in the Y-direction, KT is the number of fine meshes in the Z-direction, and NMQ is the number of input source moments. For this array, the first string is composed of the P_0 source values for each x mesh point in the first y mesh in the first z layer. The next string is the P_0 source values in the second y mesh in the first z layer. This process is repeated for all JT y meshes of the first z layer. Then repeat for each of the remaining z layers. Then starting again with the first y mesh in the first z layer, the P_1 source values for each x mesh are given. After all P_1 values are given, the P_2 values follow. Continue until all NMQ moments are specified.

Note: Usually, values for the quantities within brackets will have already been specified in the input. Sometimes, however, a quantity is derived from the array input itself. For instance, in this particular case, NMQ is not an input quantity; rather, the code counts the number of strings and then, knowing JT, KT, and NGROUP, deduces what NMQ must have been.

MINI MANUAL

Title Line Control

(5I6 Format)

NHEAD,NOTTY,NOLIST, NPASS, RESTART

Title Line(s)

(IF NHEAD>0)

Block-I:Controls & Dimensions

IGEOM
NGROUP
ISN
NISO
MT
NZONE
IM
IT
JM
JT
KM
KT
IQUAD
NN

MAXLCM
MAXSCM

FMMIX

NOSOLV
NOEDIT

NOGEOD
NOMIX
NOASG
NOMACR
NOSLNP
NOEDTT
NOADJM

T

Block-II:Geometry

XMESH [IM+1]
YMESH [JM+1]
ZMESH [KM+1]

XINTS [IM]
YINTS [JM]
ZINTS [KM]

ZONES [IM;JM*KM]
LEVELS [IM;JM*KM]
T

Block-III: Cross Sections

LIB

valid: *ODNINP*
XSLIB
ISOTXS
GRUPXS
BXSLIB
MACRXS
MACBCD
XSLIBB
(local)*MENDF*
(local)*MENDFG*
(local) *NDILIB*
alternate XSLIB name

WRITMXS

valid: *MACBCD*
XSLIBB
XSLIBF
XSLIBE

LNG

BALXS
NTICHI
CHIVEC [NGROUP]
CHIMIX
GRPSTR
LIBNAME

Rest of this block is needed only for text libraries.

MAXORD
IHM
IHT
IHS
IFIDO
ITITL
I2LP1
SAVBXS
KWIKRD (default:1)
NAMES [NISO]
EDNAME [IHT-3]
NTPI [NISO]
VEL [NGROUP]
EBOUND [NGROUP+1]

T

Block-IV: Mixing

MATLS [-;MT]
ASSIGN [-;NZONE]

PREMIX [-;-]

ASGMOD [-;-]
CMOD

MATNAM [MT]
ZONNAM [NZONE]
MATSPEC [-]

valid: *ATFRAC*
WTFRAC
ATDEN

ATWT [-]

T

Block-V: Solver

IEVT
ISCT
ITH
IBL
IBR
IBT
IBB
IBFRNT
IBBACK

EPSI
IITL
IITM
OITM
NOSIGF

iff LIB= ODNINP, insert
ASCII text cross sections here

Solver (continued)

---Accelerator Controls---

SRCACC

valid: *DSA*
TSA
NO

NORM

CHI [NGROUP;M]

DIFSOL

valid: *MG*
CG3L
CG1L
RBL
MG1L
CGMG
MGPLNZ
RBPT

TSASN

TSAEPSI

TSAITS

TSABETA

--- K-Code Convergence ---

KCALC

--- Output Controls ---

FLUXP

KPRINT

XSECTP

FISSRP

SOURCP

ANGP

BALP

RAFLUX

RMFLUX

AVATAR

WRLNK3D

ASLEFT

ASRITE

ASBOTT

ASTOP

ASFRNT

ASBACK

Solver (continued)

--- Miscellaneous ---

LSSN

LSXS

TRCOR

valid: *DIAG*
BHS
CESARO
NO

BHGT

BWTH

NORM

CHI [NGROUP;M]

DEN [IT;JT*KT]

--- or ---

DENX [IT], DENY [JT], DENZ [KT]

ANGFLX

FLXMOM

NOFXUP

--- Quadrature -----

GRPSN [NGROUP]

WGTDIA

WGT [MM]

MU [MM]

ETA [MM]

NLL [NN]

--- Transport Solver ---

TRNSOL

NODAL

WDAMP [NGROUP]

--- Flux Guess -----

INFLUX

--- Searches -----

IPVT

PV

EV

Solver (continued)

--- Searches (cont'd) ---

EVM
XLAL
XLAH
XLAX
POD
DSASRCH

XM [IM]
YM [JM]
ZM [KM]

--- Parallelization ---

NPEY
NPEZ
NCHUNK

--- Mesh Potential ---

MSHCOL
MSHCEWT [NGROUP]
MSHCASG
MSHCBNF
MSHCRBN
MSHCXVD
MSHCXRG [6]
MSHCPR
MSHCMSS
MSHFACT

--- Time Dependence ---

TIMEDEP
TIMIITL
TO
TS
DELTI
DELTMIN
INITTF
EOM
EFACT
STIMES [M]
SAMP [M]
XTIMES [N]
XAMP [N]
DTIMES [L]
NTIMES [L]
RDMPNME

Solver (continued)

----Volumetric Source----

INSORS

SOURCE [NGROUP;NMQ]

--- or ---

SOURCX [IT;NMQ] and

SOURCY [JT;NMQ] and

SOURCZ [KT;NMQ]

--- or ---

SOURCX [IT;NMQ] and

SOURCY [JT;NMQ] and

SOURCZ [KT;NMQ] and

SOURCE [NGROUP;NMQ]

--- or ---

SOURCE [IT;JT*KT*NGROUP*NMQ]

--- or ---

SOURCE [IT;JT*KT*NMQ] and

SOURCE [NGROUP;NMQ]

----Boundary Source----

BSFILE [6]

Options 1: ----

SILEFT [NGROUP;JT*KT]

SIRITE [NGROUP;JT*KT]

SIBOTT [NGROUP;IT*KT]

SITOP [NGROUP;IT*KT]

SIFRNT [NGROUP;IT*JT]

SIBACK [NGROUP;IT*JT]

Options 2: ----

SALEFT [MM*4;NGROUP*JT*KT]

SARITE [MM*4;NGROUP*JT*KT]

SABOTT [MM*4;NGROUP*IT*KT]

SATOP [MM*4;NGROUP*IT*KT]

SAFRNT [MM*4;NGROUP*IT*JT]

SABACK [MM*4;NGROUP*IT*JT]

Options 3a: ----

BSLFTG [NGROUP]

BSLFTY [JT]

BSLFTZ [KT]

BSLFTA [MM*4]

BSRITG [NGROUP]

BSRITY [JT]

BSRITZ [KT]

BSRITA [MM*4]

Solver (continued)

----Boundary Source (cont'd)----

BSBOTG [NGROUP]

BSBOTX [IT]

BSBOTZ [KT]

BSBOTA [MM*4]

BSTOPG [NGROUP]

BSTOPX [IT]

BSTOPZ [KT]

BSTOPA [MM*4]

BSFRNG [NGROUP]

BSFRNX [IT]

BSFRNY [JT]

BSFRNA [MM*4]

BSBAKG [NGROUP]

BSBAKX [IT]

BSBAKY [JT]

BSBAKA [MM*4]

Options 3b: ----

BSLFTG [NGROUP]

BSLFTYZ [JT;KT]

BSLFTA [MM*4]

BSRITG [NGROUP]

BSRITYZ [JT;KT]

BSRITA [MM*4]

BSBOTG [NGROUP]

BSBOTXZ [IT;KT]

BSBOTA [MM*4]

BSTOPG [NGROUP]

BSTOPXZ [IT;KT]

BSTOPA [MM*4]

BSFRNG [NGROUP]

BSFRNXY [IT;JT]

BSFRNA [MM*4]

BSBAKG [NGROUP]

BSBAKXY [IT;JT]

BSBAKA [MM*4]

SOLVER (continued)

----Boundary Source (cont'd)----

Options 3c: ----

BSLFTG [NGROUP]

BSLFYZ [JT;KT*MM*4]

BSRITG [NGROUP]

BSRITYZ [JT;KT*MM*4]

BSBOTG [NGROUP]

BSBOTXZ [IT;KT*MM*4]

BSTOPG [NGROUP]

BSTOPXZ [IT;KT*MM*4]

BSFRNG [NGROUP]

BSFRNXY [IT;JT*MM*4]

BSBAKG [NGROUP]

BSBAKXY [IT;JT*MM*4]

Options 3d: ----

BSLFTG [NGROUP]

SALEFT [MM*4;JT*KT]

BSRITG [NGROUP]

SARITT [MM*4;JT*KT]

BSBOTG [NGROUP]

SABOTT [MM*4;IT*KT]

BSTOPG [NGROUP]

SATOPT [MM*4;IT*KT]

BSFRNG [NGROUP]

SAFRNT [MM*4;IT*JT]

BSBAKG [NGROUP]

SABAKT [MM*4;IT*JT]

--- First Collision Source ---

FCSRC

valid: *PTANA*

SOLVER (continued)

RAYTR

PTRAY

PTBEAML

PTBEAMR

PTBEAMB

PTBEAMT

PTBEAMF

PTBEAMK

UMCFLUX

NO

FCTERM

FCNRAY

FCNTR

FCWCO

FCWCPO

FCSEED

FCRSTRT

FCXPOS

FCYPOS

FCZPOS

FCPOANG

FCAZANG

BFSIZE

T

Block-VI: EDIT

PTED
ZNEB

POINTS [K], $K \leq IT * JT * KT$
EDZONE [IT;JT*KT]

EDXS [K], $K \leq NEDT$
RESDNT
EDISOS [K], $K \leq NISO$
EDCONS [K], $K \leq NISO$
EDMATS [K], $K \leq MT$
XDF [IT]
YDF [JT]
ZDF [KT]

RSFE [NGROUP;-]
RSFX [IT;-]
RSFY [JT;-]
RSFZ [KT;-]
RSFNAM [-]

ICOLL [K], $K \leq NGROUP$
IGRPED

MICSUM [-]
IRSUMS [-]

MASSED

POWER
MEVPER

RZFLUX
RZMFLX
EDOUTF
BYVOLP
AJED
FLUXONE

PRPLTED
IPLANE
JPLANE
KPLANE

T

PARTISN INPUT DETAILS

Introduction

The following pages of this section give details for each of the input arrays. All valid PARTISN arrays are discussed in this section in detail complete enough to form the input.

However, the beginning user, particularly one unfamiliar with discrete-ordinates codes, may find that he is missing some information of a background nature. See “What Is Available Elsewhere” on page 2-14 for that.

First, here are a few general instructions:

1. All six of the input blocks are normally included. Block-I is always required but any of the other five blocks may be omitted under the proper conditions. The input module reads each block in turn and from it generates one or more binary interface files. The interface files drive the SOLVER and EDIT modules. Thus, if the user wants no edits, the Block-VI input may be omitted. Then with no interface file, the EDIT module will not be executed. Alternatively, if the interface file is available from another source, the corresponding block of input may be omitted. For instance, Block-II describes the geometry. The input module normally writes this information to the GEODST interface file. If a GEODST or a LNK3DNT file is available from another source or a previous run, the Block-II input may be omitted.
2. A general theme of the PARTISN input is that arrays that are not needed are not entered. Presence of an array indicates that it should be used. Thus, for example, if the density array is entered (DEN array), the cross section at each mesh interval will be modified accordingly. No separate switch need be set to say that the calculation should be done. To eliminate the density modification, simply remove the DEN array from the input or comment it out.
3. The arrays, in general, are grouped in the input instructions according to function. Thus, for example, the input arrays for the volumetric source are found in a single table, or grouping, of input.
4. Groupings of input data may be marked as “Required” or “Optional” in order to guide the user and speed navigation through the input instructions.

“Required” means that at least one of the arrays in the grouping must be entered. Thus, you must read through the grouping and enter at least one of the arrays found there.

Groupings marked “Optional” may be skipped if the subject is inappropriate. Thus, using the previous example, if one has no volumetric source, one simply skips to the next grouping of input; there is no need to read about any of the arrays within the volumetric source grouping.

Arrays in groupings not marked as “Required” or “Optional” should be reviewed. These groupings contain arrays of vital data that are used in every calculation, but have default values. Thus, although you may not make any input to these arrays and they are in that sense optional, you must concern yourself with them to ensure that the default values are what is intended.

5. Input arrays may also be marked individually. If not marked, they inherit the marking of the grouping in which they are contained. Thus, an unmarked array in a “Required” grouping is required input and you must enter that array. An unmarked array in an “Optional” grouping is optional.

You may encounter a “Required” array within an “Optional” grouping. That means that if you decide to invoke the option represented by that grouping, you must input that particular array. For example, if you want user defined response function reaction rates calculated, you must input the RSFE array.

All arrays within unmarked groupings are optional. However, values in these arrays may be used by the code, so you should concern yourself with the default values if you choose not to enter a value.

6. Unless specifically noted otherwise, the default on all numeric inputs is zero.
7. In an adjoint run, none of the groupwise input arrays should be inverted. The code will externally identify all groups by the physical group number, not by the calculational group number (the calculational group number is in inverse order). Thus, the user interface should be consistently in the physical group order.
8. The use of information within square brackets to indicate the size of arrays and strings and the order within those arrays is the same as described in “MINI-MANUAL Introduction” on page 2-23.
9. Except where noted, arrays and strings must contain the exact number expected by the code (as indicated in the array or string description). If not, the code will eventually abort with a (hopefully) descriptive error message or messages.
10. New users reading these instructions for the first time and unfamiliar with the PARTISN input may find it helpful to follow the sample input in Appendix A while reading this section.
11. Array names are shown here in upper case. What you should actually input for them will depend upon the code’s implementation on your platform. At the present time, on most platforms, you should use lower case input.

12. Items in italics in the input instructions indicate actual values that may be entered for an array. You will frequently find switches where the input is the digit 0 or the digit 1. This will be represented by *0/1* in the input description. In other arrays where an exact character string is required such as “ISOTXS” in the LIB array, you will find the notation *ISOTXS*. Note that in this notation, the word is both upper case and italicized. This combination means you must enter exactly those characters. Again, although the characters will be shown here in upper case, what you should actually input for them will depend upon the code’s implementation on your platform.
13. When a template for the input form is given, as for the MATLS array, the style in the template tells the user what is expected. If an input word or value is lower case and italicized, the user is to replace that position with the entry of his choice. If the input word is in italicized style and in upper case, the user is to input exactly those characters to achieve the desired result. Depending on the implementation on your platform, the input word, itself, is usually in lower case.
14. Units to be used for the input quantities are not spelled out as they only need to be self consistent. However, the following are commonly used: Dimensions in centimeters, isotopic cross sections in barns per atom; then it follows that atom densities are in atoms per barn-centimeter. Sources are particles per cm^3 per second for volumetric sources and particles per cm^2 per second for boundary sources; fluxes will then be in particles per cm^2 per second.

Title Line Details

Title Line Control (format 5I6)^a {Required}

Word	Name	Comments
1	NHEAD	Number of title lines that follow. ^b
2	NOTTY	Suppress output to on-line user terminal? <i>0/1 = no/yes.</i>
3	NOLIST	Suppress listing of all ASCII text input? <i>0/1 = no/yes.</i>
4	NPASS	Not used.
5	RESTART	Perform a time-dependent restart. <i>0/1 = no/yes.</i>

- a. WARNING! Note that this first line is in fixed format.
- b. Follow this control line with NHEAD title lines containing descriptive comments. Each title line may contain up to 72 characters.

Block-I Details: Dimensions and Controls

Dimensions {Required}

Name	Comments
IGEOM	Geometry. 1/2/3/4/6/7/11/14/15 = planar/cylindrical/spherical/two-angle slab/x-y/r-z/r-theta/x-y-z/r-z-theta or use one of the following character strings: <i>PLANE, CYLINDER, CYL, SPHERE, SPH, SLAB, SLAB2ANG,X-Y, R-Z, R-THETA, X-Y-Z, R-Z-THETA</i>
NGROUP	Number of energy groups.
ISN	S_n order to be used.
NISO	Number of physical isotopes on the basic input cross-section library.
MT	Number of physical materials ^a to be created.
NZONE	Number of geometric zones ^b in problem.
IM	Number of coarse mesh intervals ^c in the x (or r) direction.
IT	Total number of fine mesh intervals ^d in the x (or r) direction.
JM	Number of coarse mesh intervals in the y (or z) direction.
JT	Total number of fine mesh intervals in the y (or z) direction.
KM	Number of coarse mesh intervals in the z (or θ) direction.
KT	Total number of fine mesh intervals in the z (or θ) direction.

- a. Material is defined on page 2-50.
- b. Zone is defined on page 3-15.
- c. Coarse mesh is defined on page 3-15.
- d. Fine mesh is defined on page 3-15.

Block-I Details: Dimensions and Controls (Cont.)

Name	Comments																				
IQUAD	<p>Source of the quadrature constants (OPTIONAL). Enter one of the following values:</p> <table border="1"> <thead> <tr> <th data-bbox="391 548 472 579"><u>Value</u></th> <th data-bbox="537 548 691 579"><u>Description</u></th> </tr> </thead> <tbody> <tr> <td data-bbox="391 590 415 621">1</td> <td data-bbox="537 590 1300 684">Traditional built-in constants. Any even value for ISN can be used between 2 and 16 (48 for 1-D), inclusive. This is the default.</td> </tr> <tr> <td data-bbox="391 705 415 737">2</td> <td data-bbox="537 705 1300 800">For 1-D, traditional built-in DPn constants [S4, S8, S12, S16, S24, S32, S40, S48, S64, or S96]. For 2D and 3-D, same as IQUAD=1.</td> </tr> <tr> <td data-bbox="391 821 415 852">3</td> <td data-bbox="537 821 1300 915">Card input (general quadrature). ISN must be set to the number of angles/octant, NN entry is REQUIRED in Block I, and NLL entry is REQUIRED in Block V.</td> </tr> <tr> <td data-bbox="391 936 415 968">4</td> <td data-bbox="537 936 789 968">Galerkin scattering.</td> </tr> <tr> <td data-bbox="391 989 415 1020">5</td> <td data-bbox="537 989 1300 1083">Triangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.</td> </tr> <tr> <td data-bbox="391 1104 415 1136">6</td> <td data-bbox="537 1104 1300 1199">Rectangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.</td> </tr> <tr> <td data-bbox="391 1220 415 1251">7</td> <td data-bbox="537 1220 1195 1251">Get constants from SNCONS file (triangular only).</td> </tr> <tr> <td data-bbox="391 1272 415 1304">8</td> <td data-bbox="537 1272 854 1304">Square DPn Chebychev.</td> </tr> <tr> <td data-bbox="391 1325 415 1356">9</td> <td data-bbox="537 1325 1300 1419">1-D Generalized Quadrature (Even moment, triangular for cylinders).</td> </tr> </tbody> </table>	<u>Value</u>	<u>Description</u>	1	Traditional built-in constants. Any even value for ISN can be used between 2 and 16 (48 for 1-D), inclusive. This is the default.	2	For 1-D, traditional built-in DPn constants [S4, S8, S12, S16, S24, S32, S40, S48, S64, or S96]. For 2D and 3-D, same as IQUAD=1.	3	Card input (general quadrature). ISN must be set to the number of angles/octant, NN entry is REQUIRED in Block I, and NLL entry is REQUIRED in Block V.	4	Galerkin scattering.	5	Triangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.	6	Rectangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.	7	Get constants from SNCONS file (triangular only).	8	Square DPn Chebychev.	9	1-D Generalized Quadrature (Even moment, triangular for cylinders).
<u>Value</u>	<u>Description</u>																				
1	Traditional built-in constants. Any even value for ISN can be used between 2 and 16 (48 for 1-D), inclusive. This is the default.																				
2	For 1-D, traditional built-in DPn constants [S4, S8, S12, S16, S24, S32, S40, S48, S64, or S96]. For 2D and 3-D, same as IQUAD=1.																				
3	Card input (general quadrature). ISN must be set to the number of angles/octant, NN entry is REQUIRED in Block I, and NLL entry is REQUIRED in Block V.																				
4	Galerkin scattering.																				
5	Triangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.																				
6	Rectangular Chebychev-Legendre built-in set. Any even value for ISN can be used up to 50. From 50 to 100, ISN must be in multiples of 10.																				
7	Get constants from SNCONS file (triangular only).																				
8	Square DPn Chebychev.																				
9	1-D Generalized Quadrature (Even moment, triangular for cylinders).																				
NN	Number of levels/octant. REQUIRED for IQUAD=3 only, otherwise disregarded.																				

Storage Requirements {Optional}

Name	Description
MAXSCM	Length of SCM desired (default=40000 ₁₀).
MAXLCM	Length of LCM desired (default=140000 ₁₀).
	MAXSCM and MAXLCM are only used for storage in the Input and Edit modules. Solver storage is performed dynamically.

Note: The above input (Dimensions plus Storage Requirements) for Block-I will cause the code to attempt to produce a full run, subject to availability of the input normally found in the other Blocks. The controls below allow shortened print files, partial runs (say, of only the input module), or cause the code to ignore any of the other input Blocks present. For full details on their use, see “PIECEWISE EXECUTION” on page 9-21.

Fine Mesh Mixing^a {Optional}

Name	Description
FMMIX	Read the composition of each fine mesh from the binary file LNK3DNT. 0/1=no/binary file LNK3DNT.

a. x-y or x-y-z geometry only.

The fine mesh mixing algorithm is designed for a general geometry option in x-y or x-y-z geometry using a volume fraction method on the fine mesh. It implies that one has an auxiliary code which will generate the volume fraction data from a general geometry description and store it on the binary file LNK3DNT.

Run Configuration Controls {Optional}

Name	Comments
NOSOLV	Suppress solver module execution. <i>0/1=no/yes.</i>
NOEDIT	Suppress edit module execution. <i>0/1=no/yes.</i>
NOGEOD	Suppress writing GEODST file even though the geometry input (Block-II) may be present. <i>0/1=no/yes.</i>
NOMIX	Suppress writing mixing files even though the mixing input in Block-IV may be present. <i>0/1=no/yes.</i>
NOASG	Suppress writing ASGMAT file even though the assignment input in Block-IV may be present. <i>0/1=no/yes.</i>
NOMACR	Suppress writing the MACRXS file even though both Block-III and Block-IV may be present. <i>0/1=no/yes.</i>
NOSLNP	Suppress writing the SOLINP file even though Block-V may be present. <i>0/1=no/yes.</i>
NOEDTT	Suppress writing the EDITIT file even though Block-VI may be present. <i>0/1=no/yes.</i>
NOADJM	Suppress writing the ADJMAC file even though an adjoint calculation is called for. <i>0/1=no/yes.</i>

Note: Default on all these controls is no.

Block-II Details: Geometry

Geometry Arrays^a {Required}

Name	Comments
XMESH [IM+1]	x (or r) coordinates of coarse mesh edges.
YMESH [JM+1]	y (or z) coordinates of coarse mesh edges.
ZMESH [KM+1]	z (or θ) coordinates of coarse mesh edges. When r-z- θ geometry used, θ is in revolutions. For r- θ geometry, use YMESH to enter the θ coordinates.
XINTS [IM]	Number of fine meshes in each coarse x (r) mesh
YINTS [JM]	Number of fine meshes in each coarse y (z) mesh
ZINTS [KM]	Number of fine meshes in each coarse z (θ) mesh
ZONES [IM;JM*KM]	Zone number ^b for each coarse mesh. This array defines the geometric zones to which cross-section materials are assigned via the ASSIGN input array of Block-IV. The zone number must not be greater than NZONE.
LEVELS [IM;JM*KM]	Level indicators for the Block AMR solver. REQUIRED for TRNSOL=3 only, otherwise ignored. The maximum level difference between any two adjacent blocks may be no greater than one. The fine mesh size for each block will be set to 2^L , where L is the input level (greater than or equal to zero). The input fine mesh must be sized such that the number of fine meshes in each coarse mesh is a power of two, and greater than or equal to the assigned level.

a. Definitions of coarse mesh, fine mesh, and zone are given in the chapter “DETAILS OF THE BLOCK-I, GEOMETRY, AND SOLVER INPUT” starting on page 3-1. See note on units on page 2-35. The information entered in this block is written to the CCCC standard interface file GEODST.

b. A zone number of zero indicates the mesh contains a void, and no cross section will be associated with that mesh. The zero zone number is not counted in the total zone count NZONE.

Block-III Details: Nuclear Data

Nuclear Data Type and Options {Required}

Name	Comments
LIB	Name ^a and form of the cross-section data file. Enter as a data item one of the following words:
<u>Word</u>	<u>Description</u>
<i>ISOTXS^b</i>	CCCC standard isotope ordered binary cross-section file.
<i>XSLIB</i>	ASCII text library supplied in a separate file named XSLIB.
<i>ODNINP</i>	ASCII text library follows after this block of input (after the T of Block-III).
<i>GRUPXS^c</i>	CCCC standard group ordered cross-section file.
<i>BXSLIB</i>	Binary library supplied as a separate file named BXSLIB. [See “Binary Form of Card-Image Libraries (the BXSLIB file)” on page 6-12.
<i>MACRXS^d</i>	Use existing files named MACRXS for SOLVER module, SNXEDT for EDIT module. These files were created in a previous run. Under this option, any remaining Block-III input and, unless otherwise specified in Block-I, any PREMIX and MATLS input in Block-IV will be ignored.
<i>XSLIBB</i>	See “XSLIBB Card-Image Library File” on page 6-13.
<i>MACBCD</i>	ASCII form of MACRXS file.
<i>MENDF</i>	(LANL only) See “The Los Alamos MENDF Cross-Section Libraries” on page 6-14.
<i>MENDFG</i>	(LANL only) See “The Los Alamos MENDF5G Gamma Cross-Section Library” on page 6-15.
<i>NDILIB</i>	(LANL only) NDI Library.
<i>other</i>	If a word other than those listed above is entered, the code will use the file with that word as its name, provided that file exists in the users file space. Such a file must be structured as an XSLIB file.

Nuclear Data Type and Options (Cont.) {Required}

Name	Comments										
WRITMXS {optional}	Controls the code's writing certain ASCII cross-section files. ^e Enter one of the following words: <table border="0"> <thead> <tr> <th><u>Word</u></th> <th><u>Description</u></th> </tr> </thead> <tbody> <tr> <td><i>MACBCD</i></td> <td>Creates the cross-section file named MACBCD, an ASCII image of the MACRXS binary file.</td> </tr> <tr> <td><i>XSLIBB</i></td> <td>Creates the cross-section file named XSLIBB, an ASCII image of the BXSILIB binary file.</td> </tr> <tr> <td><i>XSLIBE</i></td> <td>Creates the cross-section file named XSLIBE, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBE is in Los Alamos 6E12 format (IFIDO=0).</td> </tr> <tr> <td><i>XSLIBF</i></td> <td>Creates the cross-section file named XSLIBF, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBF is in FIDO fixed-field format (IFIDO=1).</td> </tr> </tbody> </table>	<u>Word</u>	<u>Description</u>	<i>MACBCD</i>	Creates the cross-section file named MACBCD, an ASCII image of the MACRXS binary file.	<i>XSLIBB</i>	Creates the cross-section file named XSLIBB, an ASCII image of the BXSILIB binary file.	<i>XSLIBE</i>	Creates the cross-section file named XSLIBE, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBE is in Los Alamos 6E12 format (IFIDO=0).	<i>XSLIBF</i>	Creates the cross-section file named XSLIBF, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBF is in FIDO fixed-field format (IFIDO=1).
<u>Word</u>	<u>Description</u>										
<i>MACBCD</i>	Creates the cross-section file named MACBCD, an ASCII image of the MACRXS binary file.										
<i>XSLIBB</i>	Creates the cross-section file named XSLIBB, an ASCII image of the BXSILIB binary file.										
<i>XSLIBE</i>	Creates the cross-section file named XSLIBE, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBE is in Los Alamos 6E12 format (IFIDO=0).										
<i>XSLIBF</i>	Creates the cross-section file named XSLIBF, an ASCII file derived from, and corresponding to, the MACRXS binary file. XSLIBF is in FIDO fixed-field format (IFIDO=1).										
LNG {optional}	Number of the last neutron group in a coupled neutron-photon library. Used only to separate neutrons from γ s in the edits.										
BALXS {optional}	cross-section balance control. Enter one of the following values: WARNING See page 6-23 before using! <table border="0"> <thead> <tr> <th><u>Value</u></th> <th><u>Description</u></th> </tr> </thead> <tbody> <tr> <td><i>-1</i></td> <td>balance cross sections by adjusting absorption cross section.</td> </tr> <tr> <td><i>0</i></td> <td>do not balance cross sections. (default)</td> </tr> <tr> <td><i>1</i></td> <td>balance cross sections by adjusting self-scattering cross section.</td> </tr> </tbody> </table>	<u>Value</u>	<u>Description</u>	<i>-1</i>	balance cross sections by adjusting absorption cross section.	<i>0</i>	do not balance cross sections. (default)	<i>1</i>	balance cross sections by adjusting self-scattering cross section.		
<u>Value</u>	<u>Description</u>										
<i>-1</i>	balance cross sections by adjusting absorption cross section.										
<i>0</i>	do not balance cross sections. (default)										
<i>1</i>	balance cross sections by adjusting self-scattering cross section.										
NTICHI {optional}	MENDF fission fraction to be used for the problem (LANL only). <i>1/2/3</i> = Pu239/U235/U238 (default is U235). Will be overridden by any CHIVEC input described below or by any zone-dependent CHI in input Block-V.										
CHIVEC [NGROUP] {optional}	Chi vector (fission fraction born into each group). Used for every isotope. Will be overridden by any zone dependent CHI input in Block-V.										
GRPSTR	Group structure to be used for collapsing. Only valid with LIB=NDILIB. Default "lanl".										
CHIMIX	Chi mixing in each zone. <i>0/1/2</i> = none/prompt/total. REQUIRES LIB = NDILIB.										

- a. On UNIX systems, the user may specify a search path for some of these files using the environment variable SNXSPATH. See "Library Search Path" on page 2-108 for details.

- b. The CCCC standard for file ISOTXS does not allow the inclusion of the $2L+1$ term in the higher order scattering cross section. However, if you have a nonstandard file which contains the $2L+1$ term, you may override by setting $I2LP1=1$. See “Text Cross-Section Library Format” on page 2-47. PARTISN will then convert the cross sections to the appropriate internal form.
- c. The $2L+1$ term on GRUPXS is treated the same as for ISOTXS. See footnote b.
- d. In the convention used in this users guide, a MACRXS library contains “material” cross sections; all the other libraries contain “isotope” cross sections.
- e. See “COUPLED NEUTRON-GAMMA CROSS SECTIONS” on page 6-17.

Alternate Library Name {Optional}

Name	Comments
LIBNAME	Alternate name of the library file. May be used only with certain types of libraries. See Table 2.1.

The entries in the LIB input variable normally dictate both the form and the name of the cross-section library. If the user specified ISOTXS, for example, the code would look for a file named ISOTXS and expect it to be in the CCCC format for an ISOTXS file.

For some libraries, the user may specify the form in the LIB array and specify separately the name in the LIBNAME array. The libraries that can be treated this way are shown in Table 2.1.

Table 2.1 LIBNAME Availability

LIB	LIBNAME AVAILABLE?
MACRXS	No
GRUPXS	Yes
ISOTXS	Yes
BXSLIB	Yes
ODNINP	No
MACBCD	No
XSLIBB	No
MENDF ^a	Yes
MENDFG ^b	Yes
XSLIB	Yes
NDILIB ^{a,b}	Yes
other	Ignored

a. Available only at Los Alamos.

b. MENDF5 or MENDF6 only.

The BXSLIB file requires special treatment. It is normally created when the original library is a text library in the ODNINP or XSLIB form. In subsequent runs, this binary BXSLIB file may be used as the source of the cross-section data. The user may wish to save this file under another name. The program, in future runs, may then access the library for reading by using LIBNAME to specify that name.

This is a wise procedure because some cases using the BXSLIB form as input also require rewriting it in order to add new information. When this situation arises, the rewritten file is always named BXSLIB. Thus, if the original BXSLIB form library had a different name, it would be protected from being overwritten. For the remainder of the current run, the program will access the file named BXSLIB.

Text Cross-Section Library Format

{Required if LIB= XSLIB or LIB=ODNINP}

Name	Comments
MAXORD	Highest Legendre order in the scattering tables.
IHM	Number of positions (entries) in each row of the cross-section table.
IHT	Position number of the total cross section.
IHS { optional }	Position number of the self-scatter cross section. (default = IHT + 1).
IFIDO { optional }	Format of the cross-section library. -1/0/1/2 = Precision(4E18)/Los Alamos(6E12)/fixed-field FIDO/free-field.
ITITL { optional }	A title line precedes each table. 0/1 = no/yes
I2LP1 { optional }	Higher order scattering cross sections on the library contain the 2L+1 term. 0/1 = no/yes. Note: For a non-standard ISOTXS or GRUPXS that contains the 2L+1 term, enter a 1 here.
SAVBXS { optional }	Save the binary form of the ASCII text library XSLIB or ODNINP for use in a subsequent run. Saved on file BXSLIB. 0/1 = no/yes.
KWIKRD { optional }	Process fixed-field FIDO-format, ASCII text library with fast processor at the sacrifice of error checking? 0/1 = no/yes (default=yes).
NAMES [NISO] { optional }	Character name for each of the input isotopes. Can be used later in mixes. (default names are: ISO1, ISO2, ... etc.).
EDNAME [IHT-3] { optional }	Character name for each of the EDIT cross-section positions used in the cross-section edits. These are the positions before the absorption cross section in the cross-section table. (default names are: EDIT1, EDIT2,...etc.).
NTPI [NISO] { optional }	Number of Legendre scattering orders for each isotope in the library. (default=MAXORD+1 in all positions).
VEL [NGROUP] { optional }	Speeds for each group. Needed only for α calculations.
EBOUND [NGROUP+1] { optional }	Energy group boundaries.

ASCII text libraries may be entered in one of the four forms indicated by the IFIDO input. All four forms share the following features: Cross sections are entered in a table optionally preceded by a title line. A table consists of NGROUP rows of entries. Each row contains the cross sections for a single group and consists of IHM entries. The user specifies the positions in the row occupied by the total and self-scattering cross sections. Order within a row (e.g., for group g) is then as follows:

$$\dots \sigma_{\text{abs}}, \nu\sigma_f, \sigma_{\text{total}}, \dots \sigma_{g+2 \rightarrow g}, \sigma_{g+1 \rightarrow g}, \sigma_{g \rightarrow g}, \sigma_{g-1 \rightarrow g}, \sigma_{g-2 \rightarrow g}, \text{ etc.}$$

Notice that all terms in the scattering matrix are in positions relative to that of the self-scattering position and the rest of the cross sections are in positions relative to the position of the total cross section. The positions before the absorption cross section are frequently used for edit cross sections. For more detail, see “Ordering of Cross Sections Within a Cross-Section Table” on page 6-10.

Different Legendre orders are in different tables, which follow in order.

The user may order the group structure either by increasing energy or by decreasing energy. However, it is conventional and desirable for most problems to order it by decreasing energy, that is, group one is the highest energy. In that case, the scattering cross sections to the left of $\sigma_{g \rightarrow g}$ such as $\sigma_{g+1 \rightarrow g}$ are upscattering terms and the terms to the right of $\sigma_{g \rightarrow g}$ are the downscattering terms.

In the Los Alamos format, the table is entered with a standard Fortran 6E12 format.

For greater precision in your input, use the 4E18 option.

In the fixed field FIDO format that ANISN¹⁷ uses, entries are made in six twelve-column fields. Each twelve-column field is divided into three subfields, a two-column numeric field, a one-column character field, and a nine-column numeric field. See page 5-19 for details if you are not familiar with this input. The last field in each table must have the character T in the character position. No array identifier should be used. This format also restricts the usable input operators to T, *, R, -, +, and Z.

In the free field form, entries do not have to be in designated columns. Rather, the rules specified in the chapter “FREE FIELD INPUT REFERENCE” starting on page 5-1 apply. Each table in this form is also terminated with the character T. No array identifier (i.e., array name with appended equals sign) should be used.

Block-IV Details: Cross-Section Mixing

A short summary of the primary mixing arrays, MATLS and ASSIGN, is given here for quick reference. Normally, THESE TWO ARRAYS ARE REQUIRED and, in most problems, would be the only arrays in this block. Other mixing arrays are also briefly described.

There are actually several nested levels of mixing. Each level has the job of calculating values from expressions of the form: $\Sigma_g = \sum_{i=1}^k N_i \sigma_{i,g}$ for each group, g . The user's job is to input the N_i for all the k components of the mixture and to specify each component, i . Component i has the cross section, $\sigma_{i,g}$. In common usage, for the first level of mixing, $\sigma_{i,g}$ is the effective microscopic cross section, and N_i is the atom density of isotope i , and Σ_g is then the macroscopic cross section of some material. In a higher level of mixing, these materials may be homogenized into a single material by using their volume fractions for the N_i . With several nested levels, the user has a great deal of flexibility in defining what Σ_g is for that level. A more complete discussion of mixing will be found in the chapter "MATERIAL MIXING TUTORIAL" starting on page 7-1.

A discussion of cross section processing is outside the scope of this document, but it should be noted that the user needs to be aware of the processing that is inherent in the input library. For instance, for materials in which there are isotopes with cross-section resonances, self shielding of the cross sections for these isotopes may be important and this effect must have been considered in the preparation of the "effective" microscopic cross sections for these isotopes. Since the self shielding is dependent on the amounts and types of the other isotopes in the material, the "effective" cross section is strictly valid only for use in a mixture which has the same composition as was used in the self shielding calculation. If the user desires to use this same "effective" microscopic cross section in some other composition (mix) of material, it is up to the user to verify the accuracy of this approach.

Primary Mixing Arrays {Required}

Name	Description
MATLS ^a [-;MT]	Instructions for mixing “isotopes” or premixes into “materials.” See details below.
ASSIGN ^b [-;NZONE]	Assignments of materials to geometric zones. See below.
PREMIX [-;-] {optional}	Instructions for mixing “isotopes” into premixes. See below.

- The information entered in the MATLS array is written to the CCCC standard interface files NDX-SRF and ZNATDN.
- Information entered in the ASSIGN array is written to the code-dependent interface file ASGMAT.

In order to understand how cross sections are mixed and the resultant material placed in the problem, we first need a little conceptual information.

The key entities used in specifying the cross-section spatial distribution are coarse mesh, zone, isotope, and material.

The basic geometry of the problem is defined with the coarse meshes specified in Block-II. The geometric areas called zones are also defined there using the ZONES array; the ZONES array designates the zone number assigned to each coarse mesh.

Here in Block-IV, we mix cross sections and assign them to the zones created in Block-II. For the purposes of this discussion, the cross sections found on the input library belong, by definition, to “isotopes,” no matter what their true nature. These “isotopes” may then be mixed to form materials, using the MATLS array. Materials are then assigned to zones using the ASSIGN array.

MATLS input array

The general form of a MATLS mix instruction is shown below:

$$\text{MATLS} = \text{mat}_1 \text{ comp}_1 \text{ den}_1, \text{ comp}_2 \text{ den}_2, \dots \text{etc} \dots ;$$

where mat_1 is the desired character name of the first material and $\text{comp}_1, \text{comp}_2$, and so on are the character names of its components which have “densities” of, respectively, $\text{den}_1, \text{den}_2$, and so on. Additional materials (i.e., $\text{mat}_2, \text{mat}_3$, and so on up to the required

number, MT) are defined in subsequent strings. Each string may contain as many components as necessary (actual limit = 500). A component is usually an isotope from the library, but may also be a temporary material created by the PREMIX array (see below). When the component is an isotope, the den_i is commonly the atom density of the isotope in that material although other definitions exist (See MATSPEC on page 2-54).

Short form: $MATLS= ISOS$

This form specifies that there should be as many materials as isotopes and that isotope number 1 is to be used for material number 1, isotope number 2 is to be used for material number 2, and so on.

In the special case where there is only a single component in a material and its density is unity, the density entry may be omitted as in the first material below:

$$MATLS= mat_1 comp_1; mat_2 comp_2 den_2; \dots etc. \dots ;$$

ASSIGN input array

The general form of the ASSIGN instruction is shown below:

$$ASSIGN= zone_1 mat_1 vol_1, mat_2 vol_2, \dots etc. \dots ;$$

where $zone_1$ is the desired character name to be used for the first zone (the one specified with numeral 1 in the ZONES array). mat_1 , mat_2 , and so on are the character names of the materials that will be present in this zone with, respectively, the “volume fractions” vol_1 , vol_2 , and so on. Additional zones (i.e., $zone_2$, $zone_3$, and so on up to the required number, NZONE) are defined in subsequent strings. Although it is highly recommended that you use character names, here it is convenient to use the numeral for the zone name because it is the same numeral entered in the ZONES array.

Short form: $ASSIGN= MATLS$

This form specifies that there are as many zones as there are materials, and that material number 1 is to be assigned to zone number 1, material number 2 to zone number 2, and so on.

NOTE: The short form $ASSIGN=MATLS$ can not be used if you intend to use the ASGMOD input array described later in this section.

PREMIX input array

The PREMIX array forms temporary materials in a way exactly analogous to the way that permanent materials are formed in the MATLS array. The difference in treatment is

that the temporary materials created by PREMIX exist only long enough to complete the mixing; they are not available for assignment to geometric zones, nor are they available for use in material edits.

The general form of a PREMIX mix instruction is shown below:

$$\text{PREMIX} = \text{tmat}_1 \text{ comp}_1 \text{ den}_1, \text{ comp}_2 \text{ den}_2, \dots \text{etc} \dots ;$$

where tmat_1 is the character name of the first material and comp_1 , comp_2 , and so on are the character names of its components which have “densities” of, respectively, den_1 , den_2 , and so on. Additional temporary materials (i.e., tmat_2 , tmat_3 , and so on) may be defined in subsequent strings. A component may be either an isotope from the library or another temporary material created by PREMIX.

The PREMIX array is useful for organizing the mixing input. For instance, it is frequently useful to mix the cross sections for a molecule of water and then in subsequent mix instructions, to input the molecular density of water as opposed to entering the atom density for both hydrogen and oxygen. Other examples are to form average cross sections for an element composed of many isotopes, or to form full density materials and then in later mix instructions to put in the volume fraction of the full density material.

Character Names vs. Numeric Names

In the foregoing discussion, isotopes, materials, and zones were identified by their character names. Optionally, they may be referred to by their ordinal number. Thus, 2 for an isotope name would call for the second isotope on the library. However, this practice is NOT recommended.

THE CHARACTER NAME FORM IS HIGHLY RECOMMENDED. It provides the most straightforward, most readable form. If the character name form is used, the naming input arrays in the table “Miscellaneous Mixing Input” on page 2-54 are not needed.

Using the character name form in one array and the numeric name form in another array is particularly discouraged. However, should one wish to use the numeric form in the MATLS and/or ASSIGN arrays, and then subsequently associate character names with the ordinal numbers, one can use the naming arrays in the following table to do so. This situation could arise if, for some reason, one wanted to use material numbers in the MATLS array, but use character material names in the ASSIGN array.

When the library is of the MENDF form, the character names that must be used for the isotope names are discussed in “The Los Alamos MENDF Cross-Section Libraries” on page 6-14.

Mixing Array for a Concentration Search {Optional}

Name	Description
ASGMOD ^a [-;-]	C ₁ parameters used in concentration searches. See the discussion below.

- a. The information entered in the ASGMOD array is written to the ASGMAT file together with the information from the ASSIGN and CMOD arrays.

ASGMOD input array

The ASGMOD array is used in conjunction with the ASSIGN array when one wishes to vary the composition of a zone or zones in order to achieve a certain value of k_{eff} or α (i.e., in a concentration search). The concentration (or volume fraction) of material x in zone z is given by the following expression:

$$C(z,x) = C_0(z,x) + C_1(z,x)*CMOD$$

where $C_0(z,x)$ is the base concentration of material x in zone z. This is the concentration (or volume fraction) entered in the ASSIGN array for material x. In these arrays, x is not any kind of an index; correspondence is made by name, rather than by position within the array. Thus, for instance, in a problem that had ten materials, we might only assign one of them to a given zone. It would then probably be in the first position in the ASSIGN array string for that zone even though it might have been say, sixth in the list of all materials.

$C_1(z,x)$ is the corresponding entry in the ASGMOD array for material x in zone z.

CMOD is the search parameter (sometimes called search eigenvalue) that will be varied by PARTISN in order to achieve the desired k_{eff} or α value. In a search calculation, the initial value for CMOD will be the input value EV.

The general form of the ASGMOD instruction is shown below:

$$\text{ASGMOD} = \text{zone } mat_m \text{ vol}_m, mat_n \text{ vol}_n, \dots \text{etc.} \dots ;$$

where *zone* is the character name of any zone in the problem, mat_m , mat_n , and so on are the character names of any of the materials that will be present in this zone, and vol_m ,

vol_n , and so on are the C_1 values for respectively, mat_m , mat_n , and so on. Additional zones may be specified in subsequent strings. All zones do not have to appear in the ASGMOD array nor do all materials within a zone have to appear in the string for that zone.

Concentration Modifier {Optional}

Name	Description
CMOD	Concentration modifier. Input value is not used in a search. See the discussion below.

The concentration modifier, CMOD, is varied by PARTISN during a search calculation. For any other type of calculation, a value of CMOD may be input and the composition of the zones will be calculated using the expression above for $C(z,x)$.

Miscellaneous Mixing Input {Optional}

Name	Comments
MATNAM [MT]	Character material names for Materials. Used only if the mat_1 name used in the MATLS array was integer. First entry in MATNAM array is the desired character name for Material number 1, second entry is the desired character name for Material number 2, etc.
ZONNAM [NZONE]	Character zone names for Zones. Used only if the zone name entry in the ASSIGN or ASGMOD array was integer. First entry in the ZONNAM array is the desired character name for Zone number 1, second entry is the desired character name for Zone number 2, etc.

Miscellaneous Mixing Input (Cont.) {Optional}

Name	Comments
MATSPEC [\leq MT]	<p>Tells code whether material mixing in the MATLS array is in terms of atomic densities, atomic fractions, and/or weight fractions.</p> <p>Allowable entries are the words:</p> <p><i>ATDENS</i> (default) atomic densities <i>ATFRAC</i>^a atomic fractions <i>WTFRAC</i> weight fractions</p> <p>Can be input as a vector with up to MT entries (one for each Material) [See “Using Atomic Fractions or Weight Fractions (MATSPEC)” on page 7-13.] If less than MT entries are made, the last entry will be used to fill out the array to a length of MT.</p>
ATWT [$\leq 2 * \text{NISO}$] {required ^b }	<p>Atomic weights of the isotopes. If using MATSPEC=ATFRAC or WTFRAC, atomic weights must be available to the code. Entries for the ATWT array are made in pairs, as follows:</p> $\text{ATWT} = \text{iso}_1 \text{ atwt}_1 \text{ iso}_2 \text{ atwt}_2 \dots$ <p>where iso_n is the isotope name (identifier) for isotope n on the cross-section library and atwt_n is that isotope's atomic weight. [See “Using Atomic Fractions or Weight Fractions (MATSPEC)” on page 7-13].</p>

- a. ATFRAC and WTFRAC cannot be used with PREMIX.
- b. Required iff MATSPEC=ATFRAC or WTFRAC and atomic weights are not available from the input library.

Block-V Details: Solver Input

Desired Calculation {Required}

Name	Comments												
IEVT	<p>Calculation type: Enter one of the following values:.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>source</td> </tr> <tr> <td>1</td> <td>k_{eff}</td> </tr> <tr> <td>2</td> <td>α (time absorption) search</td> </tr> <tr> <td>3</td> <td>concentration search</td> </tr> <tr> <td>4</td> <td>dimension search</td> </tr> </tbody> </table>	Value	Description	0	source	1	k_{eff}	2	α (time absorption) search	3	concentration search	4	dimension search
Value	Description												
0	source												
1	k_{eff}												
2	α (time absorption) search												
3	concentration search												
4	dimension search												
ISCT	Legendre order of scattering (default = 0).												
ITH	0/1 = direct/adjoint calculation (default = 0).												
IBL	Left boundary condition ^a : 0/1/3 = vacuum/reflective/white (default = vacuum).												
IBR	Right boundary condition: 0/1/3 = vacuum/reflective/white (default = vacuum).												
IBT	Top boundary condition: 0/1/2/3 = vacuum/reflective/periodic/white. (default = vacuum).												
IBB	Bottom boundary condition: 0/1/2/3 = vacuum/reflective/periodic/white. (default = vacuum).												
IBFRNT	Front boundary condition: 0/1/2/3 = vacuum/reflective/periodic/white. (default = vacuum).												
IBBACK	Back boundary condition: 0/1/2/3 = vacuum/reflective/periodic/white. (default = vacuum).												

a. The left boundary condition applies only for slab, two-angle slab, x-y or x-y-z geometry.

Iteration Controls {Required}

Name	Comments
EPSI	Convergence precision (default=0.0001).
IITL	Maximum number of inner iterations per group at first (default chosen by code).
IITM	Maximum number of inners allowed when near fission source convergence (default chosen by code).
OITM	Maximum number of outer iterations (default=20).
NOSIGF ^a	Inhibit fission multiplication in a fixed source problem. 0/1 = no/yes.

- a. The situation envisioned by this option is that a fission problem has previously been run in which case a FIXSRC file will have been automatically written. That FIXSRC file will contain the pointwise source given by $(1/k_{eff})\chi_g \Phi$ where Φ is the fission distribution. Then the problem is rerun with NOSIGF=1 and INSORS=1 to achieve the same answer as the original. This can then be used to study differences from the original system that do not impact the fission source.

Acceleration Controls {Optional}

Name	Comments
SRCACC	Choice of transport source acceleration method. Enter one of the following (default is DSA):
	<u>Word</u> <u>Description</u>
	<i>DSA</i> Use diffusion synthetic acceleration.
	<i>TSA</i> Use transport synthetic acceleration.
	<i>NO</i> Use no source acceleration.
DIFFSOL	Choice of diffusion operator solver. Enter one of the following words (default is MG for single processor and CG1L for multi-processor):
	<u>Word</u> <u>Description</u>
	<i>MG</i> Use the multigrid solver with 3 line relaxation.
	<i>CG3L</i> Use conjugate gradient preconditioned by 3 line relaxation.
	<i>CG1L</i> Use conjugate gradient preconditioned by 1 line relaxation
<i>RBL</i> Use red-black line.	

Acceleration Controls {Optional}

Name	Comments
	<i>MGIL</i> Use the multigrid solver with 1 line relaxation.
	<i>CGMG</i> Use conjugate gradient preconditioned by multigrid with one line relaxation.
	<i>MGPLNZ</i> Use the plane-Z multigrid solver.
	<i>RBPT</i> Red black point preconditioner.
TSASN	SN order used for the low order TSA sweeps. Default is 2.
TSAEPSI	Convergence criteria for TSA sweeps. Default is 0.01.
TSAITS	Maximum number of TSA iterations. No default, except that while the fission source is not near convergence, TSAITS is set to 3.
TSABETA	Amount of reduction in scattering cross section for TSA. Default 0.0.

K-Code Convergence {Optional}

Name	Comments
KCALC	Special Criticality Convergence Scheme. 0/1 = no/yes.

A special convergence scheme may be invoked for problems which require a good eigenvalue, but do not require tight convergence of the pointwise fluxes. It consists of converging the eigenvalue, but not the pointwise fluxes. Normally both must be converged. It also sets the default for eigenvalue convergence to 0.001 rather than 0.0001. To invoke this option to save running time, set the input parameter KCALC to unity.

Output Controls {Optional}

Name	Comments						
FLUXP	Final flux print. 0/1/2 = no/isotropic/all moments.						
KPRINT	K-plane to print final fluxes on. Default KT/2 when IBFRNT=0, 1 otherwise.						
XSECTP	Cross-section print. 0/1/2 = no/principal/all.						
FISSRP	Fission source rate print. 0/1 = no/yes.						
SOURCP	Source print. 0/1/2/3 = no/as input/normalized/both.						
ANGP	Print angular flux. 0/1 = no/yes. CAUTION! This is very LARGE output. ANGP=1 will cause the RAFLXM or AAFLXM file to be written. Only valid for TRNSOL=0.						
BALP	Coarse Mesh Interval Print Options. Enter one of the following values: <table border="0" style="margin-left: 40px;"> <thead> <tr> <th><u>Value</u></th> <th><u>Description</u></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>None</td> </tr> <tr> <td>1</td> <td>Print coarse mesh balance tables.</td> </tr> </tbody> </table>	<u>Value</u>	<u>Description</u>	0	None	1	Print coarse mesh balance tables.
<u>Value</u>	<u>Description</u>						
0	None						
1	Print coarse mesh balance tables.						
RAFLUX	Prepare angular flux file (RAFLXM or AAFLXM). 0/1 = no/yes.						
RMFLUX	Prepare flux moments file (RMFLUX or AMFLUX). 0/1 = no/yes.						
AVATAR	Prepare the special ASCII file XMFLUXA. 0/1 = no/yes.						
ASLEFT	Write right-going angular edge flux at plane <i>i</i> to file bsleft.						

Output Controls (Cont.) {Optional}

Name	Comments
WRLNK3D	Write the binary LNK3DNT fine mesh geometry file with the initial mesh. 0/1 = no/yes.
ASRITE	Write left-going angular edge flux at plane <i>i</i> to file bsrite.
ASBOTT	Write top-going angular edge flux at plane <i>j</i> to file bsbot.
ASTOP	Write bottom-going angular edge flux at plane <i>j</i> to file bstop.
ASFRNT	Write back-going angular edge flux at plane <i>k</i> to file bsfrnt.
ASBACK	Write front-going angular edge flux at plane <i>k</i> to file bsback.

Miscellaneous Solver Input {Optional}

Name	Comments										
LSSN	Use the LSSN positive scattering method. 0/1=no/yes										
LSXS	Write/read the modified LSSN moments from the file lsxsdat. 0/1/2=no/write/read										
TRCOR	Apply transport correction ^a to cross sections on MACRXS file. Enter one of the following words: <table border="1" data-bbox="695 688 1393 976"> <thead> <tr> <th><u>Word</u></th> <th><u>Description</u></th> </tr> </thead> <tbody> <tr> <td><i>DIAG</i></td> <td>Use diagonal transport correction.</td> </tr> <tr> <td><i>BHS</i></td> <td>Use Bell-Hansen-Sandmeier correction.</td> </tr> <tr> <td><i>CESARO</i></td> <td>Use Cesaro "correction."</td> </tr> <tr> <td><i>NO</i></td> <td>(or omit entry) Use no correction.</td> </tr> </tbody> </table>	<u>Word</u>	<u>Description</u>	<i>DIAG</i>	Use diagonal transport correction.	<i>BHS</i>	Use Bell-Hansen-Sandmeier correction.	<i>CESARO</i>	Use Cesaro "correction."	<i>NO</i>	(or omit entry) Use no correction.
<u>Word</u>	<u>Description</u>										
<i>DIAG</i>	Use diagonal transport correction.										
<i>BHS</i>	Use Bell-Hansen-Sandmeier correction.										
<i>CESARO</i>	Use Cesaro "correction."										
<i>NO</i>	(or omit entry) Use no correction.										
BHGT	Buckling height (in cm. if macroscopic cross section in cm^{-1} .) Used only for plane, cylindrical, and two-angle plane geometries. (default=0.0, which is treated as infinity).										
BWTH	Buckling width. Used only for plane and two-angle plane geometries. (default=0.0, which is treated as infinity).										
NORM	Normalize the fission source rate to this value when IEVT.GE.1 or normalize the inhomogeneous source rate to this value when IEVT=0. NORM=0 means no normalization. (Integral of source rate over all angle, space, and energy = NORM, except for k_{eff} problems where the integral is equal to $\text{NORM} * k_{\text{eff}}$.) Any fluxes printed here (i.e., caused by setting FLUXP nonzero) will be normalized consistently with this source rate.										
CHI [NGROUP;M]	Fission fraction born into each group. ^b Enter by zone up to M zones. Succeeding zones (i.e., zones M+1 through NZONE) will use the CHI values from zone M.										
DEN [IT;JT*KT] or	Density factor to use at each fine mesh point.										
DENX [IT] ^c and/or	Density factor to use at each fine x-mesh (default=1).										

Miscellaneous Solver Input (Cont.) {Optional}

Name	Comments
DENY [JT] and/or	Density factor to use at each fine y-mesh (default=1).
DENZ [KT]	Density factor to use at each fine z-mesh (default=1).
ANGFLX	Write time-dependent angular fluxes to disk. Not operational for TRNSOL=3. 0/1=no/yes
FLXMOM	Write flux moments to disk. Not operational for TRNSOL=3. 0/1=no/yes
NOFXUP	Perform negative flux fixup. 0/1=yes/no

- For more information, see "Transport Corrections for the Cross Sections (TRCOR)" on page 3-35.
- This input will override any previous CHI from earlier blocks or from any cross-section library which contains CHI.
- In this second form, the density factor DEN(i,j,k), at mesh interval (i,j,k) is computed as follows:

$$\text{DEN}(i,j,k) = \text{DENX}(i) * \text{DENY}(j) * \text{DENZ}(k).$$

Quadrature Details {Optional}

Name	Description
GRPSN [NGROUP] ^a	S_n order to be used for each group.
WGTDIA	Use weighted diamond angular differencing (1-D spheres and cylinders only). 0/1=no/yes.
WGT [MM] ^b	Quadrature weights.
MU [MM]	Mu cosines.
ETA [MM]	Eta cosines.
NLL [NN]	Number of angles per level for general quadrature. REQUIRED for IQUAD=3.

- a. Values must be less than or equal to ISN in Block-I. The GRPSN option may not be used with boundary source input.
- b. $MM = ISN * (ISN + 2) / 8$ for $iquad = 1, 2,$ and 5 .
 $MM = (ISN / 2) ** 2$ for $iquad = 6$.

Transport Solver Details {Optional}

Name	Description
TRNSOL	Transport solver type. 0/1/2/3=vectdl/seqdp/blockj/block AMR. Default is vectdl for single processor and seqdp for multi-processor.
NODAL	Spatial differencing scheme. 0/1/2/3=standard low-order (DD,DD/STZ,AWDD)(default)/exponential discontinuous/linear discontinuous/linear discontinuous via the exponential discontinuous solver.
WDAMP[NGROUP]	Flags to activate adaptive weighted diamond differencing (AWDD) ¹⁸ for each group. 0.0/W = no/activate AWDD with parameter W. ^a If W = 0.0, the default diamond with fixup is used.

a. Recommend $1.0 < W < 4.0$ for shielding applications.

Flux Guess From a File {Optional}

Name	Comments
INFLUX	<p>Read the initial flux guess from a file.^a 0/1 = no/yes.</p> <p>If ITH=0 and ISCT>0, and the flux moments file RMFLUX exists, read the initial flux guess from RMFLUX. Otherwise, read the initial flux guess from the RTFLUX file.</p> <p>If ITH=1 and ISCT>0, and the adjoint moments file AMFLUX exists, read the initial flux guess from AMFLUX. Otherwise, read the initial flux guess from the ATFLUX file.</p>

a. There is presently no text input flux guess available for PARTISN.

General Eigenvalue Search Control^a {IEVT >1}

Name	Comments
IPVT	Type of eigenvalue to search for in a concentration or dimension search. 0/1/2 = none / k_{eff} / α . (default = 1).
PV	Value of k_{eff} or α to which to search. (default = 1.0 if IPVT=1, 0.0 if IPVT=2).
EV	Initial search parameter. Value at which to start the search parameter. (default=0).
EVM	Initial search parameter increment. Amount by which to change search parameter in the first step of a search. (default = 0.01 for DSASRCH=0 and 2.5 for DSASRCH=1).
XLAL	Lambda lower limit for search. (default = 0.01).
XLAH	Lambda upper limit for search. (default = 0.5).
XLAX	Lambda convergence criterion for second and subsequent search steps. (default = 0.001).
POD	Parameter oscillation damper. (default=1.0).
DSASRCH	Use DSA estimate of $d(\alpha) / d(\lambda)$ for alpha eigenvalue search. 0/1 = no/yes.

a. See "Eigenvalue Searches" on page 3-38 for definitions of these quantities.

PARTISN can vary the composition or dimensions of a zone (or zones) in order to achieve a desired k_{eff} or α value. The search input consists of the above general search input plus input specific to the type of search being performed.

Dimension Search Input {Required if IEVT=4}

Name	Comments
XM [IM]	x-dimension fractional change per coarse mesh.
YM [JM]	y-dimension fractional change per coarse mesh.
ZM [KM]	z-dimension fractional change per coarse mesh.

The dimension search requires the XM and/or YM and/or ZM input as well as the general search input above. During the search, PARTISN varies the search parameter (sometimes called the search eigenvalue) denoted by EV in the following expressions to change the coarse mesh boundaries:

$$\text{XMESH}_{i+1} = \text{XMESH}_i + \{\text{XMESH}_{i+1} - \text{XMESH}_i\} * [1.0 + \text{EV} * \text{XM}_i], \quad i=1, \dots, \text{IM}$$

$$\text{YMESH}_{j+1} = \text{YMESH}_j + \{\text{YMESH}_{j+1} - \text{YMESH}_j\} * [1.0 + \text{EV} * \text{YM}_j], \quad j=1, \dots, \text{JM}$$

$$\text{ZMESH}_{k+1} = \text{ZMESH}_k + \{\text{ZMESH}_{k+1} - \text{ZMESH}_k\} * [1.0 + \text{EV} * \text{ZM}_k], \quad k=1, \dots, \text{KM}$$

Although they may seem a bit awkward at first, the user will find these expressions to be quite flexible. With proper choice of the XM_i , YM_j , and ZM_k values, the user can move any or all of the coarse mesh boundaries while allowing others to remain stationary. The

quantities in { } in the above expressions are always formed from the original input values.

Concentration Search Input {Required if IEVT=3}

Name	Description
	The solver input for a concentration search is to set IEVT = 3 (page 2-56) and input the general eigenvalue search controls. But you must also input the ASGMOD ^a array in Block-IV.

- a. A concentration search involves the mixing instructions. A discussion of the ASGMOD array is found in the mixing input description on page 2-53.

Parallelization Details {Optional}

Name	Description
NPEY	Number of processors to be used for the Y-processor mesh.
NPEZ	Number of processors to be used for the Z-processor mesh.
NCHUNK	Specifies the number of X spatial planes to be solved before communicating for TRNSOL=1 (default 10). For TRNSOL=3, specifies the number of X spatial planes to be grouped together during decomposition (default 1).
The numbers of processors used is NPEY*NPEZ. If NPEY*NPEZ is a power-of-two, then entry of NPEY and NPEZ is OPTIONAL. Otherwise, entry of NPEY and NPEZ is REQUIRED.	

Mesh Potential Input {Optional}

Name	Comments
MSHCOL	Mesh potential control.
	<p>0 No Mesh collapse (default).</p> <p>1 For single-level problems, print mesh coarsening suggestions to the incol3d file and exit without solving. For Block AMR problems, print final block level suggestions to output and exit without solving.</p> <p>2 Regenerate the mesh using the mesh potential information and solve.</p>
MSHCEWT [IGM]	Energy weighting spectrum to be used with the mesh potential function. Default 1.0.
MSHCASG	Weighting parameter for the total cross section (default 0.9).
MSHCBNF	Weighting parameter for the nu*fission cross section (default 0.1).
MSHCRBN	Mesh potential value (default 0.5/NGROUP). For single-level problems, the mesh potential function will attempt to coarsen planes of cells where the calculated mesh potential (based on MSHCEWT, MSHCASG, and MSHCBNF) is less than the lower bound MSHCRBN. For Block AMR, a block's level may be reduced if the maximum potential in any cell is lower than MSHCRBN.
MSHCXVD	For single-level problems, allow coarsening of planes of void cells. For Block AMR, allow void blocks to be reduced to level 0. 0/1=yes/no.
MSHCXRG[6]	Lower and upper limits of fine mesh cells in each direction between which no coarsening is allowed (ex. 0 10 0 20 0 5). For Block AMR, MSHCXRG should be filled with the lower and upper limits of the coarse mesh region that needs protection from level reduction. All blocks in that region will be protected. The array elements correspond to the (left right bottom top front back) limits. Default 0 0 0 0 0 0.

Mesh Potential Input {Optional}

Name	Comments
MSHCPR	Include mesh potential diagnostic information in the output. Intended to help users override default mesh potential parameters when they do not result in the desired amount of coarsening. For single-level problems, the output is the maximum potential for every plane of cells in every direction. For Block AMR, the output is the maximum potential in the block containing the most important cells and in the block containing the least important cells. 0/1=no/yes.
MSHCMSS	The maximum relative change in source between neighboring cells that is allowed during coarsening (Maximum Source Slope). A negative value suppresses source control. Default 0.5.
MSHFACT	For non-AMR problems, use mshfact to ensure that the number of fine meshes contained within two adjacent coarsened mesh cells do not differ by more than a factor of mshfact. E.g., with mshfact=2, an initial incol pattern of (20 5*4 20) will become (12 8 5*4 8 12) instead. This allows for better resolution of flux gradients.

Time-Dependent Input {Optional}

Name	Comments
TIMEDEP	Run in time-dependent mode (DD/STZ in time). 0/1 = no/yes.
TIMIITL	Value of IITL to be used at the start of every cycle (default initial IITL).
T0	Initial time.
TS	Final time.
DELTI	Initial time step size.
DELTMIN	Minimum allowed time step size (default 0.0001).
INITTF	Time-dependent flux initialization. 0/1/2 = given/settle/stepstart.
EOM	The input source (for source problems) is set to this value as a floor (default 0.0).

Time-Dependent Input {Optional}

Name	Comments
EFACT	Time step controller parameter. For source only problems, if the change in the maximum pointwise total reaction rate is less than $0.2 \cdot \text{EFACT}$, then the time step is increased by 20%. For fissioning problems, if the change in the total neutron population and the change in the fission source are both less than $0.1 \cdot \text{EFACT}$, then the time step is increased by 20%. Default 0.5.
STIMES [M]	Time points at which the source is assumed linear (i.e., the source has a linear dependence between $\text{STIMES}(m)$ and $\text{STIMES}(m+1)$).
SAMP [M]	Relative value of the source amplitude at each source time point.
XTIMES [N]	Time points at which the cross section is assumed linear (i.e., the cross section has a linear dependence between $\text{XTIMES}(n)$ and $\text{XTIMES}(n+1)$).
XAMP [N]	Relative value of the cross section at each XTIMES time point. The cross section change is governed by the ASGMOD array in Block IV of the input.
DTIMES [L]	Time boundaries for dump intervals.
NTIMES [L]	Number of dumps to be (approximately) uniformly spaced within each dump interval.
RDMPNME	Dump name prefix to which the cycle number will be appended.

Volumetric Source Options {Optional}

Name	Comments
INSORS	Read source from interface file FIXSRC . $0/1$ =no/yes
---- For a text-input source, choose one of the following options:	
Option 1:	
SOURCE [NGROUP; NMQ]	Source spectrum for each of NMQ^a moments. (Spatial distribution is assumed to be flat with value unity)
Option 2:	
SOURCX [IT;NMQ] ^b	(input all three arrays) x (or r) spatial distribution for each moment.

Volumetric Source Options {Optional}

Name	Comments
SOURCY [JT;NMQ]	y (or z) spatial distribution for each moment.
SOURCZ [KT;NMQ]	z (or θ) spatial distribution for each moment.
(Spectrum is assumed to be flat with value unity)	
Option 3:	(input all four arrays)
SOURCE [NGROUP; NMQ]	Source spectrum.
SOURCX [IT;NMQ]	x (or r) spatial distribution for each moment.
SOURCY [JT;NMQ]	y (or z) spatial distribution for each moment.
SOURCZ [KT; NMQ]	z (or θ) spatial distribution for each moment.
Option 4:	
SOURCEF [IT;JT*KT*NGROUP*NMQ]	Spatial distribution for each row, group, and moment.
Option 5:	(input both arrays)
SOURCE [NGROUP; NMQ]	Source spectrum.
SOURCEF [IT;JT*KT*NMQ]	Spatial distribution for each row and moment.

- a. NMQ is not an input value but is computed from the number of strings read. NMQ must correspond exactly to the number of moments in a P_n expansion of the source. The number of moments is $(n+1)^2$ in 3-D, $(n+1)*(n+2)/2$ in 2-D, $(n+1)$ for 1-D slabs and spheres, $(n+2)^2/4$ for 1-D cylinders, and $(n+1)^2$ for two-angle slabs. n must be less than or equal to ISCT. See page 8-25 for more details.
- b. Only in option 4 is the complete pointwise source array, SOURCEF(i,j,k,g,m), given. In all other cases, it must be formed from the lower dimension arrays that are input. That calculation is done by forming the product of those arrays. Thus, in option 3, where the source spectrum, SOURCE(g,m), and the spatial distributions SOURCX(i,m), SOURCY(j,m), SOURCZ(k,m) are given (for moment m), the full source at mesh point (i,j,k) in group g for moment m is calculated as follows:

$$\text{SOURCE}(i,j,k,g,m) = \text{SOURCE}(g,m) * \text{SOURCX}(i,m) * \text{SOURCY}(j,m) * \text{SOURCZ}(k,m)$$

Boundary Source Input {Optional}

Name	Comments
<p>BSFILE[6] Use boundary source interface file for boundary sources. The six positions correspond to each boundary source file in the following order; BSLEFT BSRITE BSBOT BSTOP BSFRNT BSBACK. For example, BSFILE= 0 1 0 0 1 1 indicates to use BSRITE, BSFRNT and BSBACK boundary source interface files. 0/1=no/yes</p>	
<p>----- For a text-input source, choose one of the following options:</p>	
<p>Option 1: Isotropic Boundary Source</p>	
SILEFT [NGROUP;JT*KT]	Isotropic source on the left face. (Spectrum at each y,z mesh interval.)
SIRITE [NGROUP;JT*KT]	Isotropic source on the right face.
SIBOTT [NGROUP;IT*KT]	Isotropic source on the bottom face.
SITOP [NGROUP;IT*KT]	Isotropic source on the top face.
SIFRNT [NGROUP;IT*JT]	Isotropic source on the front face.
SIBACK [NGROUP;IT*JT]	Isotropic source on the back face.
<p>Option 2: Full Angular Boundary^a Source</p>	
SALEFT [MM ^b *4;NGROUP*JT*KT]	Angular flux on the left for each angle, group, and y,z mesh interval.
SARITE [MM*4;NGROUP*JT*KT]	Angular fluxes on the right face.
SABOTT [MM*4;NGROUP*IT*KT]	Angular fluxes on the bottom face.
SATOP [MM*4;NGROUP*IT*KT]	Angular fluxes on the top face.
SAFRNT [MM*4;NGROUP*IT*JT]	Angular fluxes on the front face.
SABACK [MM*4;NGROUP*IT*JT]	Angular fluxes on the back face.

a. The order of the angles is identical to that used in the S_n Constants table in the output file. The order of the angular octants is: $\mu < 0, \eta < 0, \tau < 0$; $\mu > 0, \eta < 0, \tau < 0$; $\mu < 0, \eta > 0, \tau < 0$; $\mu > 0, \eta > 0, \tau < 0$; $\mu < 0, \eta < 0, \tau > 0$; $\mu > 0, \eta < 0, \tau > 0$; $\mu < 0, \eta > 0, \tau > 0$, and $\mu > 0, \eta > 0, \tau > 0$, where each angular boundary source requires four octants for specification.

b. See "Quadrature Details" on page 2-63 for value of MM.

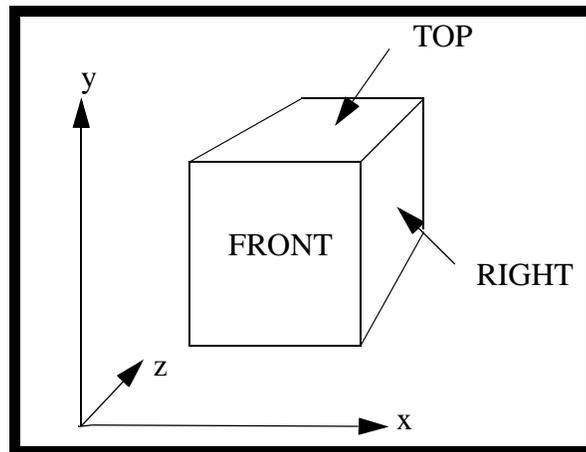


Figure 2.2 Orientation of Faces

Note the non-standard orientation of the z axis and accordingly that top is NOT in the z direction.

Within option 3 are four suboptions that involve different combinations of ‘vector’ type input for the boundary sources. All the vector input is defaulted to unity if not entered explicitly. The full angular source for the left face, for example, is constructed as follows for the 4 options (the source construction on the other faces is analogous):

$$\text{Option 3a- } S(m,g,y,z) = \text{BSLFTG}(g)*\text{BSLFTY}(y)*\text{BSLFTZ}(z)*\text{BSLFTA}(m)$$

$$\text{Option 3b- } S(m,g,y,z) = \text{BSLFTG}(g)*\text{BSLFTYZ}(y,z)*\text{BSLFTA}(m)$$

$$\text{Option 3c- } S(m,g,y,z) = \text{BSLFTG}(g)*\text{BSLFTYZ}(y,z,m)$$

$$\text{Option 3d- } S(m,g,y,z) = \text{BSLFTG}(g)*\text{SALEFT}(m,y,z)$$

Boundary Source Vector Input Combinations {Optional}

Left	Right	Bottom	Top	Front	Back
Option 3a: Boundary Source By Product Of Vectors					
BSLFTG [NGROUP]	BSRITG [NGROUP]	BSBOTG [NGROUP]	BSTOPG [NGROUP]	BSFRNG [NGROUP]	BSBAKG [NGROUP]
BSLFTY [JT]	BSRITY [JT]	BSBOTX [IT]	BSTOPX [IT]	BSFRNX [IT]	BSBAKX [IT]
BSLFTZ [KT]	BSRITZ [KT]	BSBOTZ [KT]	BSTOPZ [KT]	BSFRNY [JT]	BSBAKY [JT]
BSLFTA [MM*4]	BSRITA [MM*4]	BSBOTA [MM*4]	BSTOPA [MM*4]	BSFRNA [MM*4]	BSBAKA [MM*4]
Option 3b: Boundary Source By Product Of A Spectrum Vector, An Angular Distribution Vector, And A 2d Spatial Array					
BSLFTG [NGROUP]	BSRITG [NGROUP]	BSBOTG [NGROUP]	BSTOPG [NGROUP]	BSFRNG [NGROUP]	BSBAKG [NGROUP]
BSLFTYZ [JT;KT]	BSRITYZ [JT;KT]	BSBOTXZ [IT;KT]	BSTOPXZ [IT;KT]	BSFRNXY [IT;JT]	BSBAKXY [IT;JT]
BSLFTA [MM*4]	BSRITA [MM*4]	BSBOTA [MM*4]	BSTOPA [MM*4]	BSFRNA [MM*4]	BSBAKA [MM*4]
Option 3c: Boundary Source By Product Of A Spectrum Vector And A 3d Space-Angle Array					
BSLFTG [NGROUP]	BSRITG [NGROUP]	BSBOTG [NGROUP]	BSTOPG [NGROUP]	BSFRNG [NGROUP]	BSBAKG [NGROUP]
BSLFTYZ [JT;KT* MM*4]	BSRITYZ [JT;KT* MM*4]	BSBOTXZ [IT;KT* MM*4]	BSTOPXZ [IT;KT* MM*4]	BSFRNXY [IT;JT* MM*4]	BSBAKXY [IT;JT* MM*4]
Option 3d: Boundary Source By Product Of A Spectrum Vector And A 3d Angle-Space Array					
BSLFTG [NGROUP]	BSRITG [NGROUP]	BSBOTG [NGROUP]	BSTOPG [NGROUP]	BSFRNG [NGROUP]	BSBAKG [NGROUP]
SALEFT [MM*4; JT*KT]	SARITT [MM*4; JT*KT]	SABOTT [MM*4; IT*KT]	SATOPT [MM*4; IT*KT]	SAFRNT [MM*4; IT*JT]	SABAKT [MM*4; IT*JT]

First Collision Source Input {Optional}

Name	Comments														
FCSRC	Use First Collision Source Option. Only the PTANA and UMCFLUX options are available in parallel. Enter one of the following words:														
	<table border="0"> <thead> <tr> <th><u>Word</u></th> <th><u>Description</u></th> </tr> </thead> <tbody> <tr> <td><i>PTANA</i>^a</td> <td>Use the semi-analytic point source at position FCXPOS, FCYPOS, FCZPOS given below. x-y-z geometry only.</td> </tr> <tr> <td><i>RAYTR</i>^b</td> <td>Use the ray tracing first collision source. x-y, r-z, x-y-z, or r-z-θ geometry only.</td> </tr> <tr> <td><i>PTRAY</i></td> <td>Use ray trace point source as in <i>PTANA</i>.</td> </tr> <tr> <td><i>PTBEAML</i></td> <td>Use the point beam option on the left face of the geometry at position FCXPOS,FCZPOS; polar angle= FCPOANG and azimuthal angle= FCAZANG. A beam incident on other faces is entered the same way except the last letter is changed to R,T,B,F,K for the right, top, bottom, front, back faces respectively.</td> </tr> <tr> <td><i>UMCFLUX</i>^c</td> <td>Use the "Unde McFlux" ray tracing first collision source. x-y or x-y-z geometry only.</td> </tr> <tr> <td><i>NO</i></td> <td>(or omit entry) - don't use</td> </tr> </tbody> </table>	<u>Word</u>	<u>Description</u>	<i>PTANA</i> ^a	Use the semi-analytic point source at position FCXPOS, FCYPOS, FCZPOS given below. x-y-z geometry only.	<i>RAYTR</i> ^b	Use the ray tracing first collision source. x-y, r-z, x-y-z, or r-z- θ geometry only.	<i>PTRAY</i>	Use ray trace point source as in <i>PTANA</i> .	<i>PTBEAML</i>	Use the point beam option on the left face of the geometry at position FCXPOS,FCZPOS; polar angle= FCPOANG and azimuthal angle= FCAZANG. A beam incident on other faces is entered the same way except the last letter is changed to R,T,B,F,K for the right, top, bottom, front, back faces respectively.	<i>UMCFLUX</i> ^c	Use the "Unde McFlux" ray tracing first collision source. x-y or x-y-z geometry only.	<i>NO</i>	(or omit entry) - don't use
<u>Word</u>	<u>Description</u>														
<i>PTANA</i> ^a	Use the semi-analytic point source at position FCXPOS, FCYPOS, FCZPOS given below. x-y-z geometry only.														
<i>RAYTR</i> ^b	Use the ray tracing first collision source. x-y, r-z, x-y-z, or r-z- θ geometry only.														
<i>PTRAY</i>	Use ray trace point source as in <i>PTANA</i> .														
<i>PTBEAML</i>	Use the point beam option on the left face of the geometry at position FCXPOS,FCZPOS; polar angle= FCPOANG and azimuthal angle= FCAZANG. A beam incident on other faces is entered the same way except the last letter is changed to R,T,B,F,K for the right, top, bottom, front, back faces respectively.														
<i>UMCFLUX</i> ^c	Use the "Unde McFlux" ray tracing first collision source. x-y or x-y-z geometry only.														
<i>NO</i>	(or omit entry) - don't use														
FCTERM	Term of the transport equation the first collision source option is to be applied to. 0/1/2 = inhomogeneous source/fission source/both.														
FCNRAY	Number of ray tracings/batch to use with ray tracing first collision source. (default = 10000).														
FCNTR	Number of batches (trials) to use with ray tracing first collision source. If FCNTR is negative, read the restart file UCFLUX, and perform FCNTR additional batches of ray traces (default = 25). If this is greater than 1 and the point beam option is used, then a second collision source is done using ray tracing from the first collision source. ^d														

First Collision Source Input {Optional}

Name	Comments
FCRSTRT	If FCRSTRT=0, generate the uflux file (single-PE only) and solve. For FCRSTRT=1, read in an existing uflux file, perform FCNTR additional trials, then solve. If FCNTR=0 and FCRSTRT=1, then read in the existing uflux file and solve without performing additional trials. For FCRSTRT=2, return immediately after generating the uflux file. This allows a uflux file to be generated on 1 PE (FCRSTRT=2), then restarted and solved on multi-PE's (FCRSTRT=1 and FCNTR=0). If uflux already exists and FCRSTRT=2, read in the existing uflux file, perform FCNTR additional trials, then exit. 0/1/2=no/yes/generate the uflux file.
FCWCO	Weight cutoff for the ray tracing first collision source. When a ray has been traced a sufficient distance to attenuate its weight to less than $w_{\min} * FCWCO$, where w_{\min} is the minimum source starting weight, then that ray is terminated. Default entry is 0.0, which corresponds to a FCWCO value of $EPSI^{**2}$.
FCWCPO	FCWCO interpretation for Uncle McFlux. If 0, as described above. If 1, use FCWCO as an absolute weight cutoff.
FCSEED	Initial random number seed for ray tracing first collision sources.
FCXPOS	X position of the point source for all point source options.
FCYPOS	Y position of the point source for all point source options.
FCZPOS	Z position of the point source for all point source options.
FCPOANG	Polar angle in radians of the point beam source. ^e
FCAZANG	Azimuthal angle in radians of the point beam source. ^f
BFSIZE	Size of Uncle McFlux communication buffers.

- a. The point source will be moved within the mesh cell to the left, bottom, front corner of the cell for accuracy considerations.
- b. The ray tracing first collision source may be used with any of the volumetric or boundary source input options.
- c. The "Uncle McFlux" ray tracing first collision source may be used with volumetric source input only.
- d. If FCNTR is greater than 1, the batches will be used to provide an estimate of the maximum relative statistical error in the calculated uncollided flux for every k plane.
- e. The polar angle is measured from the positive Z axis.
- f. The azimuthal angle is measured from the positive X axis in the XY plane.

Block-VI Details: Edit Input

The input* in this block controls the calculation of reaction rates using the flux solution from the solver module.

Edit Spatial Specifications {Required^a}

Name	Comments
PTED	Do edits by fine mesh. 0/1 = no/yes.
ZNED	Do edits by zone. 0/1 = no/yes. (i.e., edit zone, not SOLVER zone. See EDZONE input below.)
POINTS [\leq IT*JT*KT] {optional}	Fine mesh point (or interval) numbers at which point edits are desired. USED ONLY IF PTED=1. (Default= all points)
EDZONE [IT;JT*KT] {optional}	Edit zone number for each fine mesh interval. USED ONLY IF ZNED=1. (default= SOLVER coarse mesh interval numbers, see ZONES array, Block-II on page 2-41)

a. Either PTED or ZNED or both must be unity in order to produce reaction rate edits.

* More details for the input for edits are given in the chapter "RUNNING THE EDIT MODULE" starting on page 4-1.

Reaction Rates from Cross Sections^a {Optional^b}

Name	Comments
EDXS [\leq NEDT] {required ^c }	<p>Cross-section types to be used in forming reaction rates.</p> <p>May be entered by integer (denoting edit position of desired cross-section type) or by the character name of the cross-section type. See the table "Edit Cross-Section Types by Position and Name" on page 2-81 or "MENDF Library Edit Cross Sections" on page 2-88 for the available names.</p> <p>NEDT is the total number of edit cross-section types available from the input cross-section library. (default = all shown in the table)</p> <p>Note: The cross-section types specified in this array apply to any or all of the following edit forms: RESDNT, EDISOS, EDCONS, EDMATS.</p>
RESDNT	Do edits using the resident macroscopic cross section at each point. 0/1 = no/yes.
EDISOS [\leq NISO]	Character names of the isotopes to be used in forming Isotopic reaction rates. The ordinal number may alternately be used but is not recommended. (default = none).
EDCONS [\leq NISO]	Character names of the isotopes to be used in forming resident Constituent (partial macroscopic) reaction rates. The ordinal number may alternately be used but is not recommended. (default = none).
EDMATS [\leq MT]	Character names of materials to be used in forming Material (macroscopic) reaction rates. The ordinal number may alternately be used, but is not recommended. (default = none).
XDF ^d [IT] YDF [JT] ZDF [KT]	Fine mesh density factors for the x(or r), y(or z) and z(or θ) directions, respectively. The density factor is used to multiply resident Constituent (see EDCONS), resident macroscopic (see EDMATS), and resident macroscopic (see RESDNT) reaction rates only. (Default= all values unity).

- See chapter "RUNNING THE EDIT MODULE" starting on page 4-1 for further discussion.
- But either something in this grouping or the next must be input in order to produce reaction rate edits.
- You must also enter one or more of the arrays EDISOS, EDCONS, EDMATS, or RESDNT.
- If density factors were used in SOLVER to modify the cross sections at each mesh interval, the same density factors must be provided here in the XDF and/or YDF and/or ZDF arrays as well. The density factor at mesh interval (i,j,k) is computed as:

$$XDF(i)*YDF(j)*ZDF(k)$$

Edit Cross-Section Types by Position and Name

CROSS-SECTION INPUT VIA ISOTXS or GRUPXS			CROSS-SECTION INPUT VIA ASCII TEXT		
<u>Type</u>	<u>EDIT Position</u>	<u>Name^a</u>	<u>Type</u>	<u>EDIT Position</u>	<u>Name</u>
chi	1	CHI....	not used	1	CHI....
nu-fission	2	NUSIGF..	nu-fission	2	NUSIGF..
total	3	TOTAL...	total	3	TOTAL...
absorption	4	ABS.....	absorption	4	ABS.....
(n,p)	5	N-PROT..	1 ^b	5	EDIT1... ^c
(n,d)	6	N-DEUT..	2	6	EDIT2...
(n,t)	7	N-TRIT..	3	7	EDIT3...
(n, α)	8	N-ALPH..	.	.	.
(n,2n)	9	N-2N....	.	.	.
(n, γ)	10	N-GAMM..	.	.	.
fission	11	N-FISS..	N=IHT-3	4+N	EDITN...
transport	12	TRNSPT..			

- Names are eight characters. A period within a name in this table denotes a blank.
- Denotes position in the cross-section table. All cross sections in positions 1 through IHT-3 in the cross-section library are EDIT cross sections chosen by the user.
- These are the default names that may be overridden with the user-option names in the EDNAME array of Block-III.

Reaction Rates from User Response Functions {Optional^a}

Name	Comments
RSFE [NGROUP;M] {required if user input response functions are desired}.	Response function energy distribution for each of the M different response functions desired. The number of different response functions is arbitrary (but must be fewer than 500). Data are entered as M strings, each with NGROUP entries beginning with group 1.
RSFX [IT;M] ^b	Response function x (or r) distribution for M functions.
RSFY [JT;M]	Response function y (or z) distribution for M functions.
RSFZ [KT;M] {optional}	Response function z (or θ) distribution for M functions. The above data are entered as M strings of IT, JT or KT entries beginning with mesh point 1. (default=1.0)
RSFNAM [M] {optional}	Character names for the user-input response functions specified above. (default = RSFP1, RSFP2,...RSFPM)

- a. But either something from this grouping or the previous one must be input in order to produce reaction rate edits.
- b. The m-th response function at space point (i,j,k) and energy group g is computed as:

$$RSFX(i,m) * RSFY(j,m) * RSFZ(k,m) * RSFE(g,m)$$

Energy Group Collapse Specifications {Optional}

Name	Comments										
ICOLL [NBG]	Edit energy group collapsing option: Number of SOLVER energy groups in each EDIT broad group. The NBG entries must sum to NGROUP. (default = 1 energy group per EDIT broad group)										
IGRPED	Print option on energy groups. Enter one of the following values: <table border="0"> <thead> <tr> <th data-bbox="597 737 670 768"><u>Value</u></th> <th data-bbox="732 737 878 768"><u>Description</u></th> </tr> </thead> <tbody> <tr> <td data-bbox="621 779 646 810">0</td> <td data-bbox="732 779 1109 810">Print energy group totals only</td> </tr> <tr> <td data-bbox="621 821 646 852">1</td> <td data-bbox="732 821 1032 852">Print broad groups only</td> </tr> <tr> <td data-bbox="621 863 646 894">2</td> <td data-bbox="732 863 1182 894">Print broad groups only (same as 1)</td> </tr> <tr> <td data-bbox="621 905 646 936">3</td> <td data-bbox="732 905 1162 936">Print both broad groups and totals</td> </tr> </tbody> </table>	<u>Value</u>	<u>Description</u>	0	Print energy group totals only	1	Print broad groups only	2	Print broad groups only (same as 1)	3	Print both broad groups and totals
<u>Value</u>	<u>Description</u>										
0	Print energy group totals only										
1	Print broad groups only										
2	Print broad groups only (same as 1)										
3	Print both broad groups and totals										

Reaction Rate Summing {Optional}

Name	Comments
MICSUM [≤ 500 sums]	<p>Cross-section reaction rate summing specifications.</p> <p>The MICSUM array is a packed array with data entered as follows: A set of Isotope numbers or names is given, followed by a set of cross-section type position numbers or names (see “Edit Cross-Section Types by Position and Name” on page 2-81). These sets are delimited with an entry of 0 (zero). Reaction rates are calculated for each Isotope specified for each cross-section type specified and summed to form the first sum. The next two sets of data are used to form the second sum, etc. Up to 500 sums can be specified. (for more detail, see “Response Function Summing Options” on page 4-13).</p>
IRSUMS [≤ 500 sums]	<p>Response function reaction rate summing specifications.</p> <p>The IRSUMS array is input as follows: A set of response function numbers or names is entered and the set delimited with an entry of 0 (zero). Reaction rates are calculated using these response functions, and the rates are summed to form the first sum. The next set of data is used to form the second sum, etc. Up to 500 sums can be specified. See page 4-13 for more detail.</p>

Mass Inventories {Optional}

Name	Comments
MASSED	<p>Calculate and print mass inventories by zone. 0/1/2/3 = none/solver zones/edit zones/both (default=1). This option is active only if atomic weights are present. See ATWT on page 2-55.</p>

Power Normalization {Optional}

Name	Comments
POWER {required}	<p>Normalize to POWER megawatts.^a</p> <p>All printed reaction rates and the fluxes on files RTFLUX and RZFLUX (if requested) will be normalized. Fluxes are normally not printed here in the EDIT module, although they may be extracted by using a unit response function. Any such fluxes will also be normalized to POWER.</p> <p>Contrast the normalization on these printed fluxes to those printed by the FLUXP input in the SOLVER Block (see NORM on page 2-61).</p>
MEVPER {required}	<p>MeV released per fission (default=210 MeV). This value will be used along with the calculated fission rate to determine the power.</p> <p>For the power calculation, PARTISN needs to know which cross section is the fission cross section. It uses the one from the library that has the name N-FISS. If one uses an ISOTXS or GRUPXS library that designation is automatically provided (See “Edit Cross-Section Types by Position and Name” on page 2-81). But if one uses an ASCII text library, either ODNINP or XSLIB, then the name N-FISS must be entered in the proper place in the EDNAME array (page 2-47).</p>

- a. Note that this normalization is meaningless if you are using the results of an adjoint run.

Miscellaneous Edit Items {Optional}

Name	Comments														
RZFLUX	Write the CCCC standard zone ^a flux file RZFLUX or AZFLUX. <i>0/1</i> = no/yes.														
RZMFLX	Write the code-dependent zone ^b flux moments file RZMFLX or AZMFLX. <i>0/1</i> = no/yes.														
EDOUTF ^c	ASCII output files control. Enter one of the following values: <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>-3</td> <td>Write both EDTOGX (without scalar fluxes) and EDTOUT files.</td> </tr> <tr> <td>-2</td> <td>Write EDTOGX file (without scalar fluxes).</td> </tr> <tr> <td>0</td> <td>Write neither file. (default)</td> </tr> <tr> <td>1</td> <td>Write EDTOUT file.</td> </tr> <tr> <td>2</td> <td>Write EDTOGX file (with scalar fluxes).</td> </tr> <tr> <td>3</td> <td>Write both EDTOGX (with scalar fluxes) and EDTOUT files.</td> </tr> </tbody> </table>	Value	Description	-3	Write both EDTOGX (without scalar fluxes) and EDTOUT files.	-2	Write EDTOGX file (without scalar fluxes).	0	Write neither file. (default)	1	Write EDTOUT file.	2	Write EDTOGX file (with scalar fluxes).	3	Write both EDTOGX (with scalar fluxes) and EDTOUT files.
Value	Description														
-3	Write both EDTOGX (without scalar fluxes) and EDTOUT files.														
-2	Write EDTOGX file (without scalar fluxes).														
0	Write neither file. (default)														
1	Write EDTOUT file.														
2	Write EDTOGX file (with scalar fluxes).														
3	Write both EDTOGX (with scalar fluxes) and EDTOUT files.														
BYVOLP	Printed point reaction rates will have been multiplied by the mesh volume. <i>0/1</i> = no/yes.														
AJED ^d	Regular (forward) edit/Adjoint edit. Regular edit uses the RTFLUX scalar flux file; adjoint edit uses the ATFLUX flux file. <i>0/1</i> = regular/adjoint.														
FLUXONE	Flux override. <i>0/1</i> = no/yes. Replaces all the input fluxes by unity. Useful for seeing the cross sections used in cross-section edits. WARNING! Meaningful reaction rates cannot be obtained when this switch is on.														

- RZFLUX and AZFLUX are organized by solver zones.
- RZMFLX and AZMFLX are organized by solver zones.
- See "ASCII File Output Capabilities (the EDOUTF Parameter)" on page 4-15.
- See "Adjoint Edits" on page 4-15.

Special Plot Linkage {Optional}

Name	Comments
PRPLTED	Write an ASCII file of the pointwise reaction rates to link to the TECPLOT [®] plotting package available commercially. -2/0/1/2/3 = 3D plots/print only/nothing/tecplot file/both print and tecplot file.
IPLANE [\leq IT]	x mesh numbers at which to write a y-z distribution.
JPLANE [\leq JT]	y mesh numbers at which to write a x-z distribution.
KPLANE [\leq KT]	z mesh numbers at which to write a x-y distribution.

To exercise this option, the user must have set PTED=1. The code will calculate reaction rates at all the fine mesh intervals, and any POINTS input will be ignored.

To link to the TECPLOT[®] code, the user chooses option 2 or 3. Separate ASCII files called rsp.dat and med.dat will be written for the response function and material edits, respectively. These files are in input form for the TECPLOT[®] preprocessor at the planes specified by IPLANE, JPLANE, and/or KPLANE.

If option 0 (print only) is chosen, no TECPLOT[®] files will be written but the reaction rates will be printed. The format of this printout is organized in a two-dimensional way unlike the normal printout from the EDIT module.

IPLANE, JPLANE, KPLANE, and POINTS input may not be used with PRPLTED=-2. The reaction ratio will be written to the file edits.ev instead of rsp.dat.

MENDF Library Edit Cross Sections

Reaction Type	Name	Description
χ	CHI	fission spectrum
$\nu\sigma_f$	NUSIGF	effective nu-sigma-fission
σ_t	TOTAL	Total cross section
σ_a	ABS	absorption ^a
(n,n)	MEND1	elastic scattering
(n,n')	MEND2	inelastic scattering
(n,2n)	MEND3	n,2n scattering
(n,3n)	MEND4	n,3n scattering
(n, γ)	MEND5	γ production
(n, α)	MEND6	α production
(n,p)	MEND7	proton production
(n,f)	MEND8	direct fission
(n,n')f	MEND9	second-chance fission
(n,2n)f	MEND10	third-chance fission
(n,F)	N-FISS	[(n,F) = (n,f) + (n,n')f + (n,2n)f]
χ_p	MEND12	prompt fission spectrum (only for fissionable materials)
χ_t	MEND13	total fission spectrum (only for fissionable materials)

a. σ_a for group g is defined as
$$\sigma_a = \sigma_t - \sum_{g'} \sigma_{g \rightarrow g'}$$

When using the Los Alamos MENDF5 cross-section library with the codes there are numerous edit cross sections available for use in the Edit Module. Since these come from the MENDF file, they are called upon with special character names in the Edit Module as part of the EDXS input. These names are defined above.

REFERENCES

1. G. I. Bell and S. Glasstone, "Discrete Ordinates and Discrete S_n Methods," in Nuclear Reactor Theory, (Van Nostrand Reinhold, New York, 1970), Chap. 5, pp. 232-235.
2. B. G. Carlson and K. D. Lathrop, "Transport Theory-Method of Discrete Ordinates," in Computing Methods in Reactor Physics, H. Greenspan, C. N. Kelber and D. Okrent, Eds. (Gordon and Breach, New York, 1968), Chap. III, p. 185.
3. "DANTSYS: A Diffusion Accelerated Neutral Particle Code System," Los Alamos National Laboratory report LA-12969-M, Los Alamos National Laboratory (1995).
4. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
5. R. E. Alcouffe, "Diffusion Synthetic Acceleration Methods for the Diamond-Difference Discrete-Ordinates Equations," Nucl. Sci. Eng. 64, 344 (1977).
6. G. L. Ramone, M. L. Adams, and P. F. Novak, "A Transport Synthetic Acceleration Method for Transport Iterations," Nucl. Sci. Eng., 125, 257 (1997).
7. R. E. Alcouffe, "The Multigrid Method for Solving the Two-Dimensional Multigroup Diffusion Equation," Proc. Am. Nucl. Soc. Top. Meeting on Advances in Reactor Computations, Salt Lake City, Utah, March 28-31, 1983, Vol. 1, pp 340-351.
8. R. S. Baker, "A Stochastic First Collision Source for TWODANT," Proc. 8th Intl. Conf. on Radiation Shielding, Arlington, Texas, April 24-28, 1994, Vol. 1, pp. 157-164.
9. J. A. Dahl, B. D. Ganapol, and J. E. Morel, "Positive Scattering Cross Sections Using Constrained Least Squares," Proc. Joint Intl. Conf. on Mathematics and Computation, Reactor Physics, and Env. Analysis in Nucl. Applications, 1, 377, Madrid, Spain (1999).
10. S. A. Turner, "Automatic Mesh Coarsening for Discrete Ordinates Codes," Proc. Joint Intl. Conf. on Mathematics and Computation, Reactor Physics, and Env. Analysis in Nucl. Applications, 2, 1423, Madrid, Spain (1999).
11. R. S. Baker, "A Block Adaptive Mesh Refinement Algorithm for the Neutral Particle Transport Equation," Nucl. Sci. Eng., 141, 1 (2002).
12. R. E. Alcouffe and R. S. Baker, "Time-Dependent Deterministic Transport on Parallel Architectures Using PARTISN," Proc. 1998 ANS Radiation Protection and Shielding Topical Conf., 1, 335, Nashville, TN (1998).
13. R. S. Baker and R. E. Alcouffe, "Parallel 3-D SN Performance for DANTSYS/MPI on the Cray T3D," Proc. of the Joint Intl. Conf. on Mathematical Methods and Supercomputing for Nucl. Applications, 1, 377, Saratoga, NY (1997).
14. R. S. Baker and K. R. Koch, "An SN Algorithm for the Massively Parallel CM-200 Computer," Nucl. Sci. Eng., 128, 312 (1998).
15. R. S. Baker, "Parallel SN Methods for Orthogonal Grids," Proc. of the 9th SIAM Conf. on Parallel Processing, San Antonio, TX (1999).

16. R. D. O'Dell and R. E. Alcouffe, "Transport Calculations for Nuclear Analysis: Theory and Guidelines for Effective Use of Transport Codes," Los Alamos National Laboratory report LA-10983-MS (September 1987).
17. W. W. Engle, Jr., "A USER'S MANUAL FOR ANISN, A One Dimensional Discrete Ordinates Transport Code With Anisotropic Scattering," Union Carbide report K-1693 (March 1967).
18. R. E. Alcouffe, "An Adaptive Weighted Diamond Differencing Method for Three-Dimensional XYZ Geometry," *Trans Am Nuc Soc.* **68**, Part A, 206 (1993).

APPENDIX A: SAMPLE INPUT

This appendix presents the printed output from a sample problem. The sample problem is a standard k_{eff} calculation with all input by means of card-images.

Sample Problem 1: Standard k_{eff} Calculation

Sample Problem 1 is a standard k_{eff} calculation for a three-dimensional model fast breeder reactor. Four energy-group cross sections are used and the scattering is assumed isotropic. Only the Input and Solver modules are executed in this sample problem, and the Solver Module is run in parallel (two processors).

The first page of the PARTISN output (page 2-97) lists the entire card-image input “deck” supplied to the PARTISN code for this sample problem. The code provides this card-image input listing unless the third entry on card 1, the entry NOLIST, is set to unity by the user. Note that numerous “comment cards” have been used in the card-image input using the slash (/) as described on page 2-20.

On page 2-98 of the problem output is a descriptive summary of the Title Card Control Parameters and a printout of the two title cards provided. This is followed by the message KEY END BLOCK-I READ, which indicates that all Block-I input has been successfully read and is ready for processing. Next appears the Block-I input summary followed by messages that both the Block-II and Block-III input card-images were successfully read.

On page 2-98 of the output is a descriptive summary of the Block-III card-image input pertaining to cross sections. Included in this summary is a listing of the cross-section types from the card-image library that can be used for edit purposes. These edit cross sections are written to the SNXEDT group-ordered cross-section interface file for use by the Edit Module, if desired. (See the tables “Edit Cross-Section Types by Position and Name” on page 2-81 and “MENDF Library Edit Cross Sections” on page 2-88.) The card-image cross-section library, provided directly in the card-image input, is read, and the header cards that were included in the library are printed for the user. For this sample problem cross sections for five isotopes have been provided. Default names are used for each isotope. The scattering is specified to be isotropic, and this is indicated by the entries “p0” under the column labeled Order. (The label “Order” refers to the Legendre order of expansion for the scattering and, since it is isotropic, only the P_0 Legendre polynomial term appears.)

On page 2-99 of the output the user is provided with a listing of all Nuclide and Material Mixing instructions provided in Block-IV of the card-image input. Isotope one (“iso1”) is used for material one, isotope two for material two, etc. Zone one is composed solely

of material one, zone two of material two, etc. Densities have already been factored into the cross sections, so no modifications are necessary. The subsequent message KEY START MIX CARD XS indicates that the PARTISN Input Module is to begin creating the working cross-section files MACRXS and SNXEDT and the standard interface files NDXSFR and ZNATDN as described in the chapter “ONEDANT, TWODANT, TWOHEX, TWODANT/GQ, and THREEDANT — Code Structure” starting on page 13-1. The last three KEY END messages on the page indicate that the cross-section mixing and processing operation was completed, the Block-V Solver Module input was read (and the SOLINP interface file created), and all Input Module operations were completed.

On page 2-99 the printed output from by the Solver Module begins. There is first presented a summary of the input parameters related to, or required by the Solver Module as provided (or defaulted). Note that for the input parameters two columns are provided: one labeled RAW INPUT and one labeled AS DEFAULTED. These columns list the values of the input parameters that the Solver Module actually uses. For example, under the heading Required Input, the value for the parameter IBR is listed as 0. (In the actual card-image input, no entry for IBR has been provided.) The default value of IBR (0.0) is, accordingly, assumed by the Solver Module and this value is provided under the AS DEFAULTED column.

On page 2-100 through page 2-101 of the output are more of the Solver input parameters and also listed are the Block-I input parameters that are carried over for use by the Solver Module. Here, for example, is indicated that the problem is x-y-z geometry (IGEOM= 14), four energy groups (NGROUP= 4), and S_8 quadrature is to be used (ISN= 8), etc.

On page 2-101 of the output is provided a recap of the assignment of materials to zones in terms of the algorithm described on page 2-53 under the ASGMOD ARRAY description in Block-IV. Below this is a summary of the discrete-ordinates quadrature quantities used for the calculation. For this problem the values printed are built-in S_8 quadrature values. Following this on the next page is a map of the problem geometry showing the coarse-mesh boundary locations, the zone number assigned to each coarse-mesh interval, and other pertinent information.

Page 2-102 lists the material names of materials for which cross-section data exist on the MACRXS interface file being used by the Solver Module. Next is provided a listing of the ZONE macroscopic cross sections used by the Solver Module. This print is optional and is controlled by the XSECTP entry in the Block-V input. In this sample the full table of ZONE macroscopic cross sections has been requested by setting XSECTP= 2. The PRINCIPAL CROSS SECTIONS are defined as the ZONE macroscopic values of χ (fission fraction), $\nu\sigma_f$, σ_p , and σ_a . * The scattering matrix terms correspond to the coef-

ficients in a Legendre expansion of the term $\sigma_{s, g' \rightarrow g}(r, \mu_0)$ in Eq. (2) on page 12-11. The value of the Legendre order for the term is provided under the column labeled ORDER in the printout. The actual scatter matrix terms for scatter from energy-group h to energy-group g are listed across the page in the sequence

$$\sigma_{s, h \rightarrow g} \sigma_{s, (h-1) \rightarrow g} \sigma_{s, (h-2) \rightarrow g}, \text{ etc.}$$

The entries in the column labeled FIRST GRP in the printout give the value of the energy-group h, namely the first group in the listing which scatters into group g. For downscatter only problems, the value of h is the same as the group number g. For upscatter problems the value of h will not be the same as the value of g. At the bottom of the output page is information related to mesh coarsening, which is not used in this problem.

The Solver Module memory requirements are output in the storage map on page 2-103. The maximum number of 8-byte words used per processor (PE) is listed, as well as the individual module component requirements. The diffusion 3d and block module component requirements are per spatial cell, while the remainder are per PE. Information on the parallel layout of the problem is also contained in this section including the number of PE's used, the spatial load balancing (ALB), and the parallel computational efficiency (PCE).¹³⁻¹⁵

On page 2-104 is provided a summary description of iteration control criteria followed by the iteration monitor print. These items are fully described on page 7-19. It is noted that for this type of problem, a k_{eff} calculation, the eigenvalue is the value of k_{eff} . For the sample problem, then, $k_{eff} = 0.97346344$ is provided in the monitor print for outer iteration 5 under the column labeled K-EFF EIGENVALUE.

Then page 2-105 provides a balance table print for each energy group and the sum of the groups. The group-dependent quantities are defined and computed as follows:

(1) SOURCE = total inhomogeneous source = QG_g

$$QG_g = \sum_{i=1} Q_i V_i + \sum_{\mu_m < 0} w_m |\mu_m| A_{IT+1/2} QR_m + \sum_{\mu_m > 0} w_m \mu_m A_{1/2} QL_m,$$

* Note that in this discussion, as in the chapter "ONEDANT, TWODANT, TWOHEX, TWODANT/GQ, and THREEDANT — Methods Manual" starting on page 12-1, a lower case sigma is used to represent a macroscopic cross section.

where Q_i is the inhomogeneous distributed source, QL_m is the left boundary (surface) source, QR_m is the right boundary (surface) source, V_i is the “volume” of spatial mesh interval i , $A_{IT+1/2}$ is the surface area at the rightmost boundary of the system, and $A_{1/2}$ is the surface area at the leftmost boundary of the system (similar boundary source terms exist for the top, bottom, front, and back faces);

(2) FISSION SOURCE = total fission source to the group g = FG_g

For a k_{eff} eigenvalue problem,

$$FG_g = \frac{1}{k_{eff}} \sum_{h=1}^{NGROUP} \sum_{i=1}^{IT} \chi_{g,i} (\nu \sigma_f)_{h,i} \phi_{h,i} V_i ;$$

For a source with fission problem,

$$FG_g = \sum_{h=1}^{NGROUP} \sum_{i=1}^{IT} \chi_{g,i} (\nu \sigma_f)_{h,i} \phi_{h,i} V_i ;$$

(3) IN SCATTER = in scatter source to group g from other groups = SIN_g

$$SIN_g = \sum_{\substack{h=1 \\ h \neq g}}^{NGROUP} \sum_{i=1}^{IT} (\sigma_{s,h \rightarrow g})_i \phi_{h,i} V_i ;$$

(4) SELF SCATTER = self-scatter (within group scatter) in group g = SS_g

$$SS_g = \sum_{i=1}^{IT} (\sigma_{s,g \rightarrow g}^o)_i \phi_{g,i} V_i ;$$

(5) OUT SCATTER = out-scatter from group g to all other groups = $SOUT_g$

$$SOUT_g = \sum_{i=1}^{IT} (\sigma'_{t,g})_i \phi_{g,i} V_i - ABG_g - SS_g ;$$

where $\sigma'_{t,g}$ is the total cross section for group g plus any buckling “absorption” plus any “time absorption” (α/v_g);

(6) ABSORPTION = absorption in group g = ABG_g

$$ABG_g = \sum_{i=1}^{IT} (\sigma'_{a,g})_i \phi_{g,i} V_i ;$$

where $\sigma'_{a,g}$ is the absorption cross section for group g plus any buckling “absorption” plus any “time absorption” (α/v_g);

(7) RIGHT LEAKAGE = net flow out of system right boundary = RL_g (similar terms are used for the other faces)

$$RL_g = \sum_{\mu_m > 0} w_m \mu_m A_{IT+1/2} \Psi_{m, IT+1/2} - \sum_{\mu_m < 0} w_m |\mu_m| A_{IT+1/2} \Psi_{m, IT+1/2} ;$$

(8) NET LEAKAGE = net flow from system (all boundaries) = NL_g

(9) PARTICLE BALANCE = BAL_g

$$BAL_g = 1 - \frac{NL_g + ABG_g + SOUT_g}{QG_g + FG_g + SIN_g};$$

(10) NPROD = spectrum of neutrons causing fissions = $NPROD_g$

$$NPROD_g = \frac{1}{N} \sum_{i=1}^{IT} (v\sigma_f)_{g,i} \phi_{g,i} V_i ;$$

$$\text{where - } N = \sum_{g=1}^{NGROUP} \sum_{i=1}^{IT} (v\sigma_f)_{g,i} \phi_{g,i} V_i .$$

(11) TIME SOURCE = incoming source at time-edge (time-dependent calculations only)

Following this is the timing history of the run.

The final page (page 2-106), provides the RUN HIGHLIGHTS for the sample problem execution.

It is to be noted that no Edit Module output appears in the output of this sample problem. The reason for this is that there is no edit input available. No Edit Module input (Block-VI of the card-image input) is provided in the input “deck” and no EDITIT

binary interface file (containing previously created Edit Module input) was in existence at the time of the sample problem execution.

Sample Problem: Output Listing

```

1
*****
*****
*
*          generalized input module run on 08/22/02 with solver version 07-25-02   beta release 2.92   machine thepolic
*
*****
*
*          ...listing of cards in the input stream...
*
*
*      1.          2          0
*      2. 3D model of a small FBR (Benchmark prob from Japan).
*      3. Cross sections from input...
*      4. /
*      5. // block 1
*      6. /
*      7. igeom=x-y-z ngroup=4 ism=8 iquad=1
*      8. nisc=5 mt=5 nzone=5
*      9. im=8 it=14 jm=8 jt=14 km=4 kt=30
*     10. maxscm=300000 maxlcm=600000
*     11. T
*     12. /
*     13. // block 2
*     14. /
*     15. xmesh=0.0 15.0 30.0 35.0 40.0 45.0 50.0 55.0 70.0
*     16. xints=3 3 1 1 1 1 1 3
*     17. ymesh=0.0 5.0 15.0 30.0 40.0 45.0 50.0 55.0 70.0
*     18. yints=1 2 3 2 1 1 1 3
*     19. zmesh=0.0 20.0 75.0 130.0 150.0
*     20. zints=4 11 11 4
*     21. zones=3r2 2r3 2r2 5; 7r2 5; 6r2 2r5; 5r2 3r5; 4r2 4r5; 2r2 6r5;
*     22.          2 7r5; 8r5;
*     23.          3r1 2r3 2r1 5; 7r1 5; 6r1 2r5; 5r1 3r5; 4r1 4r5; 2r1 6r5;
*     24.          1 7r5; 8r5;
*     25.          3r1 2r3 2r1 5; 7r1 5; 6r1 2r5; 5r1 3r5; 4r1 4r5; 2r1 6r5;
*     26.          1 7r5; 8r5;
*     27.          3r2 2r3 2r2 5; 7r2 5; 6r2 2r5; 5r2 3r5; 4r2 4r5; 2r2 6r5;
*     28.          2 7r5; 8r5;
*     29. T
*     30. /
*     31. // block 3
*     32. /
*     33. lib=cdnimp
*     34. ihm=8 iht=4 ihs=5          / 1 activity position in slot 1
*     35. ititl=1 ifido=2
*     36. /
*     37. // inline cross sections in free-form FIDO format follow this block
*     38. /
*     39. T
*     40. core
*     41. 1. 0.00745551 0.0206063 0.114568 0.0704326 3r0.0 /
*     42. 1. 0.0035254 0.00610571 0.205177 0.195443 0.0347967 2r0.0 /
*     43. 1. 0.00780136 0.00691403 0.329381 0.320586 0.00620863 0.00188282 0.0 /
*     44. 1. 0.0274496 0.0260689 0.38981 0.36236 9.92975e-4 7.07208e-7 0.0 /
*     45. T
*     46. axblkt
*     47. 1. 0.00535418 0.013177 0.116493 0.0716044 3r0.0 /
*     48. 1. 0.00148604 1.26026e-4 0.220521 0.210436 0.037317 0.0 0.0 /
*     49. 1. 0.005353 1.5238e-4 0.344544 0.337506 0.00859855 0.00221707 0.0 /
*     50. 1. 0.0134694 7.87302e-4 0.388356 0.374886 0.0016853 6.68299e-7 0.0 /
*     51. T
*     52. macrod
*     53. 1. 3.10744e-4 0.0 0.0658979 0.0474407 3r0.0 /
*     54. 1. 1.13062e-4 0.0 0.10981 0.106142 0.0176894 0.0 0.0 /
*     55. 1. 4.48988e-4 0.0 0.186765 0.185304 0.00355466 4.57012e-4 0.0 /
*     56. 1. 0.00107518 0.0 0.209933 0.208858 0.0010128 1.77599e-7 0.0 /
*     57. T
*     58. cntrod
*     59. 1. 0.00597638 0.0 0.184333 0.134373 3r0.0 /
*     60. 1. 0.0176941 0.0 0.366121 0.318582 0.0437775 0.0 0.0 /
*     61. 1. 0.0882741 0.0 0.615527 0.519591 0.0298432 2.06054e-4 0.0 /
*     62. 1. 0.476591 0.0 1.09486 0.618265 0.00766209 8.71188e-7 0.0 /
*     63. T
*     64. radblk
*     65. 1. 0.00743283 0.0189496 0.119648 0.0691158 3r0.0 /
*     66. 1. 0.0019906 1.75265e-4 0.242195 0.230626 0.0404132 0.0 0.0 /
*     67. 1. 0.00679036 2.06978e-4 0.356476 0.348414 0.00957027 0.00268621 0.0 /
*     68. 1. 0.0158015 0.00113451 0.379433 0.363631 0.00127195 1.99571e-7 0.0 /
*     69. T
*     70. /
*     71. // block 4
*     72. /
*     73. matls=isos
*     74. assign=matls
*     75. T
*     76. /
*     77. // block 5
*     78. /
*     79. ievt=1 ibl=1 ibb=1 ibfmt=0 norm=1 iitl=1 oitm=11 xsectp=2
*     80. epsi=1.0e-3 chi=0.583319 0.40545 0.011231 0.0
*     81. kcalc=1 nchunk=7
*     82. T
*****
1
*****
*****
*
*          case title
*
*****
*****
*key start case input *
*****

```

```

*
*          2 nhead  number of title cards to follow
*          0 notty  0/1 no/yes suppress on-line terminal output
*          0 nolist 0/1 no/yes suppress input listing
*          0 restart 0/1 no/yes read td_restart file
*
*          *****
*          * 3D model of a small FBR (Benchmark prob from Japan).
*          * Cross sections from input...
*          *
*          *****
*****
*key end  block i read*
*****
1
*****
*
*          ..block i - controls and dimensions...
*
*          ..dimensions ...
*
*          14 igeom  14/15 x-y-z/r-z-theta
*          4  ngroup number of energy groups
*          8  isn    angular quadrature order
*          1  iquad  1/2/3/4/5/6/7/8/9 - source of quadrature constants (default=1)
*                   1 traditional built-in Pn set (1-D) or old twotran built-in set
*                   2 traditional built-in DPN constants (1-D), tri DPN-Tchev product (2/3-D)
*                   3 card input
*                   4 galerkin (based on triangular hybrid product set)
*                   5 hybrid product set (triangular arrangement)
*                   6 product set (rectangular arrangement)
*                   7 srcon file
*                   8 DPN-Tchebychev (sq product set)
*                   9 Generalized Quadrature
*          5  niso  number of input isotopes (from isotxs, groups, or cards)
*          5  mt   number of permanent materials
*          5  nzone number of zones
*          8  im   number of coarse mesh x intervals
*          14 it  number of fine mesh intervals
*          8  jm   number of coarse mesh y intervals
*          14 jt  number of fine mesh y intervals
*          4  km   number of coarse mesh z intervals
*          30 kt  number of fine mesh z intervals
*
*          ..storage...
*
*          maxlcm= 600000
*          maxscm= 300000
*
*          *****
*          *key end  block ii read-geom*
*          *****
*
*          14 igeom  1/2/3/6/7/11/14/15
*          *****
*key end  block iii read-xs *
*****
1
*****
*
*          ..block iii - cross section library...
*
*          ..library source...
*          lib=cdnmp
*
*          ..card library parameters (array name = cards)...
*
*          0 maxord maximum legendre order to be found in input cross sections
*          8 ilm   last position in cross section table
*          4 iht   position of total cross section
*          5 ihs   position of self scatter cross section
*          2 ifido -1/0/1/2 - dtf(4e18,0)/dtf/fixed fido/free fido library
*          1 ititl 0/1 - no/yes there is a title card before each table
*          0 i2lpl 0/1 - no/yes library higher order scattering contains 2l+1 factor
*          0 savbxs 0/1 - no/yes save binary xslib file (filename=bxslib)
*          0 kwikrd 0/1 - full fido read/quick fido read (default=quick)
*
*          ..energy structure...
*
*          group   chi       vel       lower bound  upper bound  group   chi       vel       lower bound  upper bound
*          -----
*          1  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  3  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
*          2  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00  4  0.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
*
*          last neutron group (lng) is number  4
*
*          0 balxs -1/0/1 - adjust absorption/no/adjust self scatter to force xs balance
*
*          ..edit position names...
*
*          card
*          position edname position
*          -----
*          1  chi
*          2  nusigf  3
*          3  total  4

```



```

* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 1 3 3 1 5 5 5
* material map for k mesh intervals 16, to 26 and j mesh intervals 14, to 1, and i mesh intervals 1, to 14.
*
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 1 1 1 5 5 5 5 5 5 5 5 5 5 5 5
* 1 1 1 1 1 5 5 5 5 5 5 5 5 5 5
* 1 1 1 1 1 1 5 5 5 5 5 5 5 5 5
* 1 1 1 1 1 1 1 5 5 5 5 5 5 5 5
* 1 1 1 1 1 1 1 1 5 5 5 5 5 5 5
* 1 1 1 1 1 1 1 1 1 5 5 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 5 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 1 1 1 1 5 5 5 5
* 1 1 1 1 1 1 1 3 3 1 5 5 5
1* material map for k mesh intervals 27, to 30 and j mesh intervals 14, to 1, and i mesh intervals 1, to 14.
*
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
* 2 2 2 5 5 5 5 5 5 5 5 5 5 5 5
* 2 2 2 2 2 5 5 5 5 5 5 5 5 5 5
* 2 2 2 2 2 2 5 5 5 5 5 5 5 5 5
* 2 2 2 2 2 2 2 5 5 5 5 5 5 5 5
* 2 2 2 2 2 2 2 2 5 5 5 5 5 5 5
* 2 2 2 2 2 2 2 2 2 5 5 5 5 5 5
* 2 2 2 2 2 2 2 2 2 2 5 5 5 5 5
* 2 2 2 2 2 2 2 2 2 2 2 5 5 5 5
* 2 2 2 2 2 2 2 2 2 2 2 2 5 5 5
* 2 2 2 2 2 2 2 3 3 2 5 5 5
*
*****
*****
1*****
*
*
*           Three-dimensional coarse mesh geometry edit...
*
*****
*key start geometry edit *
*****
* n   xmesh   xint  deltax   ymesh   yint  deltax   zmesh   zint  deltax
*
* 0  0.00E+00   3  5.00E+00   0.00E+00   1  5.00E+00   0.00E+00   0.00E+00   4  5.00E+00
* 1  1.50E+01   3  5.00E+00   5.00E+00   2  5.00E+00   7.50E+01   11  5.00E+00
* 2  3.00E+01   1  5.00E+00   1.50E+01   3  5.00E+00   1.30E+02   11  5.00E+00
* 3  3.50E+01   1  5.00E+00   3.00E+01   2  5.00E+00   1.50E+02   4  5.00E+00
* 4  4.00E+01   1  5.00E+00   4.00E+01   1  5.00E+00   4.50E+01   1  5.00E+00
* 5  4.50E+01   1  5.00E+00   5.00E+01   1  5.00E+00   5.50E+01   1  5.00E+00
* 6  5.00E+01   1  5.00E+00   5.50E+01   1  5.00E+00   7.00E+01   3  5.00E+00
* 7  5.50E+01   1  5.00E+00   7.00E+01   3  5.00E+00
* 8  7.00E+01   3  5.00E+00   7.00E+01
*
*****
1*****
*
*
*           ...cross section related data from file macrxs 134518082220 version 1 ...
*
*****
* 1 iso1   2 iso2   3 iso3   4 iso4   5 iso5
*
*****
1*****
*
*
*           ...cross sections for legendre orders up to p 0...
*
*****
*key start mac cross sections*
*****
***** group 1 *****
*
*
*           ...principal cross sections...
*
*
*           zone      chi      nu*fission      total      absorption
*           no.  name
* 1  zone1  5.8332E-01  2.0606E-02  1.1457E-01  7.4555E-03
* 2  zone2  5.8332E-01  1.3177E-02  1.1649E-01  5.3542E-03
* 3  zone3  5.8332E-01  0.0000E+00  6.5898E-02  3.1074E-04
* 4  zone4  5.8332E-01  0.0000E+00  1.8433E-01  5.9764E-03
* 5  zone5  5.8332E-01  1.8950E-02  1.1965E-01  7.4328E-03
*
*
*           ...scattering matrices...
*           (2l+1 not included)
*
* zone  order  first grp  cross sections
* 1     0     1       7.0433E-02
* 2     0     1       7.1604E-02
* 3     0     1       4.7441E-02

```

```

*   4   0   1   1.3437E-01
*   5   0   1   6.9116E-02
*
***** group 2 *****
*
*           ..principal cross sections..
*
*           zone      chi      nu*fission      total      absorption
*           no.  name
*   1  zone1  4.0545E-01  6.1057E-03  2.0518E-01  3.5254E-03
*   2  zone2  4.0545E-01  1.2603E-04  2.2052E-01  1.4860E-03
*   3  zone3  4.0545E-01  0.0000E+00  1.0981E-01  1.1306E-04
*   4  zone4  4.0545E-01  0.0000E+00  3.6612E-01  1.7694E-02
*   5  zone5  4.0545E-01  1.7527E-04  2.4220E-01  1.9906E-03
*
*           ..scattering matrices...
*           (2l+1 not included)
*
*   zone  order  first grp  cross sections
*   1     0     2          1.9544E-01  3.4797E-02
*   2     0     2          2.1044E-01  3.7317E-02
*   3     0     2          1.0614E-01  1.7689E-02
*   4     0     2          3.1858E-01  4.3778E-02
*   5     0     2          2.3063E-01  4.0413E-02
*
***** group 3 *****
*
*           ..principal cross sections..
*
*           zone      chi      nu*fission      total      absorption
*           no.  name
*   1  zone1  1.1231E-02  6.9140E-03  3.2938E-01  7.8014E-03
*   2  zone2  1.1231E-02  1.5238E-04  3.4454E-01  5.3530E-03
*   3  zone3  1.1231E-02  0.0000E+00  1.8677E-01  4.4899E-04
*   4  zone4  1.1231E-02  0.0000E+00  6.1553E-01  8.8274E-02
*   5  zone5  1.1231E-02  2.0698E-04  3.5648E-01  6.7904E-03
*
*           ..scattering matrices...
*           (2l+1 not included)
*
*   zone  order  first grp  cross sections
*   1     0     3          3.2059E-01  6.2086E-03  1.8828E-03
*   2     0     3          3.3751E-01  8.5985E-03  2.2171E-03
*   3     0     3          1.8530E-01  3.5547E-03  4.5701E-04
*   4     0     3          5.1959E-01  2.9843E-02  2.0605E-04
*   5     0     3          3.4841E-01  9.5703E-03  2.6862E-03
*
***** group 4 *****
*
*           ..principal cross sections..
*
*           zone      chi      nu*fission      total      absorption
*           no.  name
*   1  zone1  0.0000E+00  2.6069E-02  3.8981E-01  2.7450E-02
*   2  zone2  0.0000E+00  7.8730E-04  3.8836E-01  1.3469E-02
*   3  zone3  0.0000E+00  0.0000E+00  2.0993E-01  1.0752E-03
*   4  zone4  0.0000E+00  0.0000E+00  1.0949E+00  4.7659E-01
*   5  zone5  0.0000E+00  1.1345E-03  3.7943E-01  1.5802E-02
*
*           ..scattering matrices...
*           (2l+1 not included)
*
*   zone  order  first grp  cross sections
*   1     0     4          3.6236E-01  9.9298E-04  7.0721E-07
*   2     0     4          3.7489E-01  1.6853E-03  6.6830E-07
*   3     0     4          2.0886E-01  1.0128E-03  1.7760E-07
*   4     0     4          6.1827E-01  7.6621E-03  8.7119E-07
*   5     0     4          3.6363E-01  1.2719E-03  1.9957E-07
*
*****
1*****
*
*key start mesh potential info *
*****
* weight factor a:          9.00000E-01
* weight factor b:          1.00000E-01
*
* energy weighting function:  2.50000E-01
*                             2.50000E-01
*                             2.50000E-01
*                             2.50000E-01
*
* user input min potential:   1.25000E-01
* user input max source slope: 5.00000E-01
*
* initial max aspect ratio:  1.00000E+00
*
*****
1*****
*
*key start storage map *
*****
* partsn storage summary...
*
*   max words/PE allocated in alloc3d for this problem  237683
*
*   words/PE for the geom3d module          322
*   words/PE for the snpr3d module          198
*   words/PE for the fcsr3d module          0
*   words/PE for the bal3d module           110
*   words/PE for the transport module        40
*   words/PE for the xsec module            217
*   words/PE for the time module            0

```

```

* max words/PE for the meshpot3d module      340
* words/cell for the diffusion3d module      41.2
* words/cell for the block module           34.0
*
* max cells/PE      2940
* min cells/PE      2940
* total cells       5880
*
* number of PE's used = 2
* PCE = 0.982, SN = 8
* ALB = 1.000
*
*****
1
*****
...iteration controls and criteria...
*****
***iteration criteria***
transport inners
-----
criterion  quantity to test      value  action taken if value exceeded
-----
iitl - imer iteration count until near lambda
      (i.e. fission source) convergence      1      terminates inners
iitm - imer iteration count when near lambda
      (i.e. fission source) convergence      30     terminates inners
epsi - fractional ptwise flux change
      per inner      1.00E-03  does another inner
*****
diffusion sub-outers
-----
criterion  quantity to test      value  action taken if value exceeded
-----
oitmd - sub-outer iteration count      22     terminates sub-outers
eps - diffusion lambda-1.0 (see note below) 1.00E-03  does another sub-outer
eps - fractional ptwise fission change
      per sub-outer (see note below) 1.00E-03  does another sub-outer
note: eps, when the problem is finally converged, will equal epsi, the value shown above. however,
early in the iteration process, a larger value may be used to avoid unnecessary iterations.
*****
final convergence criteria
-----
criterion  quantity to test      value  action taken if value exceeded
-----
oitm - outer iteration count      11     quits with error message
epsi - transport lambda-1.0      1.00E-03  does another outer
*****
1
*****
...flux and eigenvalue convergence as monitored by partism...
*****
*key start iteration monitor *
*****
cpu time outer      diffusion      k-eff      max ptwise      max ptwise      inners
(sec) no. inners sub-outers eigenvalue lambda-1 flux change fiss change converged
* 0.40 0
* 1.15 1 0 5 0.94625959 8.71687E-03 0.00000E+00 0.00000E+00 **no**
* 3.11 2 4 2 0.97183687 2.04534E-02 2.41730E-01 2.60224E-01 **no**
* 5.06 3 4 2 0.97317648 4.44334E-04 3.73616E-02 3.59170E-02 **no**
* 7.00 4 4 2 0.97341081 5.81321E-05 6.55255E-03 6.50322E-03 **no**
*
-----
-- inner iteration summary for outer iteration no. 5 --
*
* iter per max flux at
* group group change mesh
* 1 1 0.12E-02 5, 12, 25
* 2 1 0.71E-03 8, 11, 25
* 3 1 0.44E-03 9, 3, 30
* 4 1 0.25E-02 9, 3, 25
*
cpu time outer      diffusion      k-eff      max ptwise      max ptwise      inners
(sec) no. inners sub-outers eigenvalue lambda-1 flux change fiss change converged
* 8.83 5 4 2 0.97346344 1.47826E-05 2.45206E-03 1.35800E-04 **no**
*
particle balance = -5.53556E-06 total inners all outers = 16
*****
1
*****
...group edit and balances upon convergence...
*****
*
*
* ..title--- 3D model of a small FBR (Benchmark prcb from Japan). ...
*
*

```

```

*
*                                     ...system balance tables... (neutrons only)
*
*
*****
*key start balance table *
*****
*
*
* gp      source      fission source      absorption      in scatter      self scatter      out scatter      net leakage
*
* 1  0.000000E+00      5.8331900E-01      9.3605396E-02      4.4408921E-16      9.0151653E-01      4.7787507E-01      1.1841037E-02
* 2  0.000000E+00      4.0545000E-01      2.4700661E-01      4.5277270E-01      1.5724706E+01      5.3335668E-01      7.7863085E-02
* 3  0.000000E+00      1.1231000E-02      4.3337483E-01      5.5837083E-01      1.9303965E+01      6.5330568E-02      7.0894661E-02
* 4  0.000000E+00      0.0000000E+00      6.0224027E-02      6.5357953E-02      9.8798048E-01      1.2076901E-06      5.1322510E-03
*
* tot  0.000000E+00      1.0000000E+00      8.3421087E-01      1.0765015E+00      3.6918168E+01      1.0765635E+00      1.6573103E-01
*
*
*
* gp      right leakage  horizontl leakage      top leakage  vertical leakage      back leakage  fr-back leakage  particle balance
*
* 1  3.9867873E-03      3.9867873E-03      4.1668781E-03      4.1668781E-03      1.8436947E-03      3.6873716E-03      -4.2927836E-06
* 2  2.5616525E-02      2.5616525E-02      2.6517731E-02      2.6517731E-02      1.2864473E-02      2.5728829E-02      -7.1184843E-06
* 3  2.3916514E-02      2.3916514E-02      2.4463514E-02      2.4463514E-02      1.1257364E-02      2.2514632E-02      -4.9155671E-06
* 4  1.5828358E-03      1.5828358E-03      1.6027472E-03      1.6027472E-03      9.7333810E-04      1.9466680E-03      -1.2451109E-06
*
* tot  5.5102663E-02      5.5102663E-02      5.6750870E-02      5.6750870E-02      2.6938870E-02      5.3877500E-02      -5.5355618E-06
*
*
*
* gp      left leakage  bottom leakage      front leakage  nprod spectrum      time source
*
* 1  0.0000000E+00      0.0000000E+00      1.8436769E-03      2.6283436E-01      0.0000000E+00
* 2  0.0000000E+00      0.0000000E+00      1.2864356E-02      3.9058318E-01      0.0000000E+00
* 3  0.0000000E+00      -3.4694470E-18      1.1257268E-02      3.0268793E-01      0.0000000E+00
* 4  0.0000000E+00      0.0000000E+00      9.7332993E-04      4.3894530E-02      0.0000000E+00
*
* tot  0.0000000E+00      -3.4694470E-18      2.6938630E-02      1.0000000E+00      0.0000000E+00
*
*****
1
*****
*
* Multigrid work units... Total= 584.08 WU.
* By group...
* 1 121.86 2 160.51 3 160.51 4 141.19
*
* Multigrid average convergence rate by group...
* 1 0.5118 2 0.5384 3 0.5134 4 0.5800
*
integral summary information
summary integral-k-eff 9.7346344E-01
summary integral-neutrons 1.5361768E+02
integral-source-i neutron 0.0000000E+00
integral-time source-i neutron 0.0000000E+00
integral-fission-i neutron 1.0000000E+00
integral-absorption-i neutron 8.3421087E-01
integral-in-scak-i neutron 1.0765015E+00
integral-self-scak-i neutron 3.6918168E+01
integral-out-scak-i neutron 1.0765635E+00
integral-net lkage-i neutron 1.6573103E-01
integral-right lkage-i neutron 5.5102663E-02
integral-horizontal lkage-i neutron 5.5102663E-02
integral-top lkage-i neutron 5.6750870E-02
integral-vertical lkage-i neutron 5.6750870E-02
integral-back lkage-i neutron 2.6938870E-02
integral-fr-back lkage-i neutron 5.3877500E-02
integral-particle bal-i neutron -1.7571946E-05
*
*
*                                     ...interface file rflux written..
*
*
*                                     ...interface file snoods written..
*
*****
PARTISN solver iteration time = 8.505 seconds
mesh analysis and adjustments = 0.003 seconds
first collision source = 0.000 seconds
outer source setup = 0.013 seconds
inner source setup = 0.011 seconds
transport sweeps = 5.813 seconds
angular source = 0.038 seconds
flux moments = 0.038 seconds
transport synthetic acceleration = 0.000 seconds
diffusion synthetic acceleration = 2.564 seconds
diffusion source = 0.053 seconds
diffusion solve = 1.423 seconds
relaxation = 0.852 seconds
interpolation = 0.133 seconds
condensation = 0.172 seconds
converge test = 0.000 seconds
diffusion setup = 1.033 seconds
particle balance = 0.007 seconds

Transport Grind Time = 788.981 nanoseconds
Diffusion Grind Time = 746.653 nanoseconds
*
*****
1
*****
*
*

```


APPENDIX B: OPERATING SYSTEM SPECIFICS

UNIX/LINUX/UNICOS Execution

On UNIX, LINUX or UNICOS systems, the input is on STDIN and the printed output is on STDOUT. Thus, the user will normally cause execution of the program with the command:

$$\text{partisn} \quad < \quad \text{odninp} \quad > \quad \text{odnout}$$

where - *partisn* is the name of the executable file, *odninp* is the user's choice for a name for the input file, and *odnout* is the user's named output file. Whoever forms the executable names the executable file. The name customarily used is *partisn*.

STDERR contains a summary of the problem as it executes and, by default, is sent to the terminal screen. Also included on STDERR are any error messages.

Library Search Path

Most files read or written by PARTISN are in the current UNIX working directory. Some forms of cross-section files may be kept in other directories. By setting the environment variable **SNXSPATH**, the user may specify an ordered set of alternate directories in which the program should look for the named files. As an example, if an ISOTXS file is in the directory, **/usr/tmp/xs**, then the following command can be used

```
setenv SNXSPATH /usr/tmp/xs
```

and PARTISN will then look in that named directory for the library. The search path for each of the possible libraries is given in Table 2.2.

Table 2.2 UNIX Search Path

LIB	SEARCH PATH
MACRXS	Current Working Directory (CWD).
GRUPXS	SNXSPATH , then CWD.
ISOTXS	SNXSPATH , then CWD.
BXSLIB	SNXSPATH , then CWD, but see text below.
ODNINP	None, the library is contained in the input file.
MACBCD	CWD
XSLIBB	CWD
MENDF ^a	Path defined in the code on UNICOS. MENDF binaries are unavailable for other platforms.
MENDFG ^a	
XSLIB	SNXSPATH , then CWD
other	For any name other than those above, the program will assume the form is XSLIB and search for it in SNXSPATH , then CWD.

a. Available only at Los Alamos.

SNXSPATH can be used to protect an input BXSLIB file from being overwritten. See the discussion on page 2-46.

PARTISN CHAPTER INDEX

A

Adaptive weighted diamond differencing	2-57
Adjoint	
Edit of	2-75
Group order in	2-32
Angular flux	
File output	2-55
Input of boundary	2-63
Print output	2-55
Array	
Definition of	2-19
Notation for order	2-22
Notation for size	2-22
ASSIGN	
Input template	2-48

B

Block	
Definition of	2-20
Order in input	2-17
Block-I input	
PARTISN	2-35
Block-II input	
PARTISN	2-38
Block-III input	
PARTISN	2-39
Block-IV input	
PARTISN	2-46
Blocks	
Input order	2-17
Block-V input	
PARTISN	2-53
Block-VI input	
PARTISN	2-68
Boundary condition	
Options	2-53
Boundary source	2-63

C

Character names	
In mixing arrays	2-49
Chi	

Choice from MENDF	2-40
Isotope independent input	2-40
Zone dependent	2-56
Comments	
Embedded in input lines	2-20
Convergence controls	
Special for criticality	2-55
Cross section library	
Balancing of	2-40
Converting to ASCII	2-40
Edit names on MENDF	2-77
Edit positions and names on	2-70
Specifying what, where	2-39
Text, format of	2-44
Text, order within	2-45
Text, position in input stream	2-17
Transport correcting	2-56
Writing ASCII library from code	2-40

D

Data item	
Character, definition of	2-19
Numeric, definition of	2-19
Data operators	
Purpose of	2-20
Summary table of	2-21
Usage form	2-20
Density factors	
In edits	2-69
In the flux calculation	2-56
Documentation available	
For PARTISN	2-13

E

Edit names on MENDF	2-77
Edit positions and names	
Table of	2-70
Edits	
Energy specifications	2-72
Reaction rates	
From cross sections	2-69
From user defined response functions	2-71
Spatial specifications	2-68
Energy groups	
Broad groups in edits	2-72
Order in adjoint run	2-32

Execution of code	
On UNIX systems	2-103
<u>F</u>	
Files	
Output, control of	2-55, 2-75
Suppressing writing	2-37
Fine mesh mixing option	2-36
First collision source options	
Ray tracing	2-66
Fission fraction	
Choice from MENDF	2-40
Isotope independent input	2-40
Zone dependent	2-56
Flux	
Input guess from file	2-59
Moments file output	2-55
Normalization of	2-56
Print control	2-55
Free field input	
Summary	2-19
<u>G</u>	
Geometry	
Input arrays	2-38
Group collapse	
In edits	2-72
Group dependent quadrature	2-58
<u>I</u>	
Input instructions	
Block order	2-17
Rules for use of	2-31
Isotope	
Concept/Definition of	2-47
Iteration controls	
Eigenvalue calculations	2-54
Of searches	2-59
Source calculations	2-54
<u>M</u>	
Marking on input arrays	
Optional marking	2-31
Required marking	2-31
Materials	
Concept/Definition of	2-47

MATLS	
Input template	2-47
MINI-MANUAL	
Definition and purpose	2-22
Graphic of	2-23
Mixing	
Concepts in	2-47
Fine mesh input option	2-36
Materials	2-47
Premixes	2-47
Modules	
Suppressing execution of	2-37
<u>N</u>	
Normalization	
Of edit reaction rates	2-74
Of the flux	2-56
Of the source rate	2-56
Notation	
For expected order within array	2-22
For expected size of array	2-22
Numeric names	
In mixing arrays	2-49
<u>O</u>	
Operators	
In input, see also "Data operators"	2-20
Summary table of	2-21
Optional marking on input arrays	2-31
Output	
Angular flux file	2-55
Angular flux print	2-55
Controls	2-55
Printed, control of contents	2-55
<u>P</u>	
PREMIX	
Input template	2-48
Purpose of	2-48
Printed output	
Control of contents	2-55
<u>Q</u>	
Quadrature sets	
Group dependent	2-58
PARTISN, choices of	2-58

R

Required marking on input arrays	2-31
Response function	
Edits of	2-71

S

Sample problem	
PARTISN	
Description	2-81
Input	2-83
Output	2-81
Searches	
Concentration	
Mixing arrays required	2-49, 2-51
Solver arrays required	2-60
Dimension, arrays required	2-60
General control of	2-59
Sn order	
Group dependent	2-58
Source	
Boundary angular fluxes	2-63
First collision	2-66
Inhibit fission multiplication	2-54
Normalization of	2-56
Volumetric, input of	2-61
Storage	
Memory requirements	2-36
String	
Definition of	2-20
Summing	
Of reaction rates	2-73
Over energy in edits	2-72

T

Terminating run	2-54
Time limit	2-54
Transport correction of cross section library	2-56

U

User's Guide contents	
PARTISN briefing	2-13

V

Void, specifying a	2-38
Volumetric source, input of	2-61

X

XSLIB

Format of 2-44

Z

Zone

Assigning a material to 2-47

Concept/Definition of 2-47

Specifying a void in 2-38

DETAILS OF THE BLOCK-I, GEOMETRY, AND SOLVER INPUT

Transport Methods Group, CCS-4
Los Alamos National Laboratory

3

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	3-3
LIST OF FIGURES	3-5
LIST OF TABLES.....	3-7
INTRODUCTION	3-9
MORE DETAILS ON BLOCK-I INPUT	3-11
Angular Quadrature (ISN).....	3-11
Geometry (NZONE, IM, IT)	3-11
MAXSCM, MAXLCM	3-12
Execution/File Suppression Flags	3-13
MORE DETAILS ON GEOMETRY INPUT	3-15
MORE DETAILS ON SOLVER INPUT	3-17
Iteration Strategy	3-17
Convergence Criteria.....	3-23
Inner Iteration Convergence.....	3-23
Diffusion Sub-Outer Iteration Convergence.	3-23
Transport Source Iteration Convergence.....	3-24
Final Convergence.....	3-24
Iteration Monitor Print	3-25
General Aspects of the Monitor Print.....	3-25
Warning Messages and Their Meanings	3-26
Boundary Conditions.....	3-26
Input of Quadrature Sets	3-27
Zone-Dependent Fission Fractions (the CHI Array).....	3-28
Input of Inhomogeneous Sources	3-30
Distributed Source Input.....	3-30
Surface (Boundary) Source Input.....	3-32
Normalization of the Calculation (the NORM Parameter)	3-35
Transport Corrections for the Cross Sections (TRCOR)	3-35
Buckling Corrections.....	3-37
Eigenvalue Searches.....	3-38
Adjoint Computations	3-41
REFERENCES	3-43
INPUT DETAILS CHAPTER INDEX	3-44

LIST OF FIGURES

Figure 3.1: Simplified flow diagram of SOLVER iteration strategy.....	3-22
Figure 3.2: Ordering in slab geometry.....	3-33
Figure 3.3: Quadrature points in cylindrical geometry.....	3-34
Figure 3.4: Variation of λ during a hypothetical eigenvalue search.....	3-41

LIST OF TABLES

Table 3.1: Source Ordering Index in Slab Geometry.....	3-33
Table 3.2: Source Ordering Index in Cylindrical Geometry.....	3-34



INTRODUCTION

This chapter is intended to provide a more detailed description of a portion of the input than is provided in the User's Guide chapter. In particular, the input to Blocks-I, II, and V will be discussed here. Detailed discussion of the input for Blocks-III, IV, and VI are found in other separate support chapters.

In order to condense the discussion and make it simpler to understand, much of the description is couched in terms of one-dimensional geometry. The extension to two and three dimensions is quite straightforward and is briefly mentioned.

MORE DETAILS ON BLOCK-I INPUT

The input parameters provided in Block-I are used by the code to determine storage requirements for the problem being run (the code uses variable dimensioning), to provide error checking on subsequent input, and/or to control the execution-flow of the code.

In this section are provided details on certain of the parameters that appear in the Block-I input for PARTISN. In some cases, the “details” are simply references to other chapters of this manual.

Angular Quadrature (ISN)

The numerical value (an even integer) entered for the parameter ISN is simply the value of N in S_n , that is, the angular quadrature order desired for the current calculation. The discrete-ordinates, or S_n , approximation is described in “Discrete-Ordinates Equations in One Dimension” on page 8-29. See “Input of Quadrature Sets” on page 3-27 for details on angular quadrature sets supplied in the code

Geometry (NZONE, IM, IT)

The parameter NZONE is the number of different zones that are to be defined for the calculation. See “MORE DETAILS ON GEOMETRY INPUT” on page 3-15 for the concept and meaning of the term zone.

The number of coarse spatial mesh intervals in the x direction of the problem being solved is denoted by the parameter IM. See “MORE DETAILS ON GEOMETRY INPUT” on page 3-15 for the concept and meaning of the term coarse mesh interval. Correspondingly, the number of coarse spatial mesh intervals in the y and z directions are denoted respectively by the parameters JM and KM.

The total number of fine spatial mesh intervals (or mesh points) for the problem being solved is given by the parameters IT for the x direction, JT for the y direction, and KT for the z direction. The fine mesh interval or point is described in “Discretization of the Spatial Variable” on page 8-35. See page 3-15 for the concept and meaning of the term fine mesh interval.

MAXSCM, MAXLCM

Originally designed for the CDC-7600 computer, the Input and Edit Modules are structured for a three-level hierarchy of data storage: a small, fast core central memory (SCM), a fast-access, peripheral large core memory (LCM), and random-access peripheral storage. (For computing systems based on a two-level hierarchy of data storage -- a large fast core and random-access peripheral storage -- a portion of fast core is designated as a simulated LCM to mimic the three-level hierarchy). Random-access storage will be automatically used by the code if LCM (or simulated LCM) storage requirements are exceeded.

The MAXSCM parameter allows the user to specify the size of SCM that is desired. The code requires a certain amount of SCM for execution. The default value of MAXSCM is $40,000_{10}$ words, a sufficiently large value to handle the majority of one dimensional problems. For the higher dimension solvers this amount may not be enough and the code will inform the user of the amount needed.

Through the use of the input parameter, MAXLCM, in Block-I of the card-image input, the user can specify the maximum amount of large core memory (LCM) he wishes to use. If unspecified, the value of MAXLCM is defaulted to $140,000_{10}$ words.

The modular structure of PARTISN is such that in the processing of each input Block, as well as in the execution of the Edit Module, each uses LCM storage independently and each such stage requires a different amount of LCM. (In most cases, the cross-section processing stage requires the greatest amount of LCM.) At each stage, the amount of LCM required for that stage is computed and compared to MAXLCM. If the amount of LCM exceeds MAXLCM, the code will automatically attempt to page LCM information to random disk, such that the stage requires no more than MAXLCM words of LCM at any one time. After Block-I is read, the sum of MAXSCM and MAXLCM is used to obtain an area from the HEAP to be used as the storage for the problem. Again, the code will inform the user of the amount being used and whether information is being paged to disk.

The user must be cautioned against specifying too small a value of MAXLCM since the result may be an excessive use of random disk, the access to which is relatively time-consuming. Also, if the user is computing on, say, a virtual memory machine with no random disk, the value of MAXLCM must be large enough so that the problem can be run without random disk.

MAXSCM and MAXLCM storage is only active during execution of the Input and Edit Modules. This storage is released (deallocated) during execution of the Solver Module. The Solver Module does not use SCM and LCM for storage. Instead, F90 ALLOCATES/DEALLOCATE are used for array storage. An option is provided through the

Block V input keywords ANGFLX and FLXMOM to store the time-dependent angular fluxes and the flux moments on disk, respectively.

Execution/File Suppression Flags

Included in the Block-I input parameters are several flags which control the execution-flow of the code or the creation of interface files of by the code. These flags are relatively specialized and are normally of interest only to the more advanced user. Accordingly, details on the use of these parameters will not be described here but are provided in another chapter. See “Module Execution Control” on page 9-21.

MORE DETAILS ON GEOMETRY INPUT

Geometry-related information is passed to the SOLVER module and the EDIT module by means of the GEODST interface file¹ or the LNK3DNT file. If no GEODST file exists prior to the execution of the code package, the user may instruct the Input Module to create the desired GEODST file by (i) providing Block-II input data in the card-image input file, and (ii) setting (or defaulting) the Block-I input parameter, NOGEOD, to zero. If, on the other hand, a pre-existing GEODST file is to be used, the user needs to suppress creation of a new GEODST which would overwrite the existing GEODST file. The user may so instruct the code by either (i) omitting all Block-II input from the card-image input file or (ii) setting the Block-I input parameter NOGEOD to unity.

In the specification of geometry and space-variable related input, the user must be familiar with the nomenclature common to these codes. The terms fine mesh, coarse mesh, and zones are defined below. The term region is not used directly in any PARTISN input and the user preparing geometry input in Block-II need not be concerned with that term. However, region is a concept used in the GEODST standard file and should the GEODST file be produced from some other code, it is possible that part of the geometry description will be in terms of regions. In any case, PARTISN will accept any GEODST file that obeys the interface file description.

The fine mesh is the spatial solution-mesh for the problem, as described in the chapter “PARTISN — METHODS MANUAL” starting on page 8-1. Each fine mesh, or fine mesh interval, is bounded by an adjacent pair of fine-mesh grid-lines. In the x direction, these are denoted $x_{i-1/2}$ and $x_{i+1/2}$ with $x_{i-1/2} < x_{i+1/2}$. There are IT, JT, and KT such fine mesh intervals in respectively the x, y, and z directions. No material discontinuities may occur within a fine mesh interval. The specification of the fine mesh is accomplished by specifying how many equally sized fine mesh intervals there are in each coarse mesh.

The coarse mesh is a spatial superset of the fine mesh and is formed by partitioning the spatial domain of the problem into a suitable number of “coarse” intervals. There are IM, JM, and KM coarse mesh intervals spanning the problem in each of the directions. Each coarse mesh interval is bounded by an adjacent pair of coarse-mesh boundaries that are specified in the input either as the XMESH array in Block-II or as the XMESH array on a GEODST standard interface file. Similarly the arrays YMESH and ZMESH are used for the other directions as appropriate. Each coarse mesh interval contains one or more fine mesh intervals. The number of fine mesh intervals per coarse mesh interval is specified by means of either the XINTS array in input Block-II or the IFINTS array on a GEODST file. The input arrays YINTS and ZINTS are used for the other directions.

All fine mesh intervals within a coarse mesh interval have equal widths. No material discontinuities may occur within a coarse mesh interval.

The region is a spatial superset of coarse mesh intervals or, conversely, a spatial subset of zone. A region contains one or more coarse mesh intervals and one or more regions comprise a zone. No material discontinuities occur within a region. The concept of the region is used only in conjunction with input from a GEODST standard interface file. For geometry input through Block-II card images, the user need not be concerned with the term. However, when the code uses the Block-II information to write a standard GEODST file, the term region is treated synonymously with the term coarse mesh interval.

The zone is a spatial superset of coarse mesh intervals and is characterized by a single set of multigroup nuclear properties, i.e., cross sections, so that all fine mesh intervals within a zone have the same cross sections. A zone number is assigned to each coarse mesh interval by either (i) the ZONES array in input Block-II, or (ii) the NZNR and MR arrays on a GEODST standard file. In the ZONES array input the zone number, n ($1 \leq n \leq \text{NZONE}$), is determined by the order in which zones are specified in the ASSIGN array input in Block-IV (See “ASSIGNING MATERIALS TO ZONES” on page 7-11), so that the zone number tells the code which macroscopic cross-section set is to be used within that zone. Coarse mesh intervals having the same zone number need not be simply connected.

In the ZONES array, the number 0 (zero) can be used to specify that a coarse mesh interval is a pure void (all cross sections are identically zero). A “0” does not count as a zone in determining the value of NZONE.

MORE DETAILS ON SOLVER INPUT

Iteration Strategy

As described in “Iteration Procedure and Diffusion Synthetic Acceleration” on page 8-14 of this document, the PARTISN solver module uses the diffusion synthetic method to accelerate the iterative procedure used in solving the transport equation. In this section is described the implementation of the iterative strategy and acceleration method in the solver module and reflected in the iteration monitor printout supplied as printed output. Also described are the input convergence controls by which the user controls the iteration process.

The basic features of the iteration strategy are shown in the simplified flow diagram of Figure 3.1, “Simplified flow diagram of SOLVER iteration strategy.,” on page 3-22. As indicated, there are two different iterative procedures, one for problems containing fissionable material and/or energy-group upscattering and one for problems with neither fissions nor upscattering.

The iterative strategy is divided into two parts: inner iterations and outer iterations. The inner iterations are concerned with the convergence of the within group scattering source and the calculation of the pointwise angular fluxes in each group. The outer iterations are concerned with the convergence of the eigenvalue, the fission source distribution, and the energy-group upscatter source if any or all are present.

For problems containing fissionable material, the iterative procedure begins with a multigroup diffusion calculation where the diffusion coefficient for each space-energy point is evaluated as:

$$D(x, g) = \begin{cases} \frac{1}{3\Sigma_{trans}(x, g)} & \text{If } \Sigma_{trans}(x, g) \text{ is available.} \\ \frac{1}{3\Sigma_t(x, g)} & \text{If not,} \\ & \text{Isotropic scatter} \\ \frac{1}{3[\Sigma_t(x, g) - \Sigma_{s1}(x, g \rightarrow g)]} & \text{Anisotropic scatter} \end{cases} \quad (1)$$

where $D(x, g)$ is the diffusion coefficient at position x for energy group g , $\Sigma_t(x, g)$ is the macroscopic total cross section at the space energy point in question, $\Sigma_{trans}(x, g)$ is the macroscopic transport cross section (transport is normally provided on the ISOTXS or GRUPXS files), and $\Sigma_{s1}(x, g \rightarrow g)$ is the P_1 anisotropic self-scatter cross section. It should be noted that $\Sigma_t(x, g)$ is formed from the isotope cross sections contained in the total cross-section position in the cross-section library. The data provided in this position may, in fact, be the transport cross section in transport-corrected cross-section libraries for isotropic scatter.

Assuming that the problem contains fissions and the default setting of the input variable, IITL, is taken (IITL=1), then the iteration procedure for solving the transport equation begins with solving the conventional, multigroup diffusion equation:

$$-\nabla \cdot D_g \nabla \phi_g^{k+1} + \sigma_{R,g}(r) \phi_g^{k+1}(r) = \frac{\chi_g}{k_{eff}} \Phi^k(r) + \sum_{g'=1}^{g-1} \sigma_{s,g' \rightarrow g} \phi_{g'}^{k+1}(r) + \sum_{\substack{g'=g+1 \\ g=1, \dots, G}} \sigma_{s,g' \rightarrow g} \phi_{g'}^k(r) \quad (2)$$

where g is the energy group number,

k is the diffusion sub-outer iteration index,

$\sigma_{R,g} = \sigma_{t,g} - \sigma_{s,g \rightarrow g}$ is the removal cross section,

$$\Phi^k(r) = \sum_{g'=1}^G (v\sigma_f)_{g'} \phi_{g'}^k(r) \quad \text{is the fission distribution.}$$

On the right hand side of Eq. (2), the first term is the fission source, the second is the down-scatter source, and the third is the upscatter source. The iteration process starts ($k=0$) by setting the flux to zero for each group but assuming a spatially flat fission distribution. Equation (2) is then solved by inverting the diffusion operator for each group on the given source, updating the down-scatter source as indicated. The inversion is by line inversion in 1D and by a multigrid procedure in 2- and 3D. Once all the groups are solved, the fission distribution is recalculated using this flux, and the upscatter source is computed if present. This iteration is called a diffusion sub-outer and continues until the convergence criterion is satisfied. This criterion is explained below in Eqs. (6) and (7).

The solution of the transport problem now continues with the inner iteration of each group of the transport equation. This iteration is written as:

$$\begin{aligned}
\underline{\Omega} \cdot \nabla \psi_g^{l+1/2}(\underline{r}, \underline{\Omega}) + \sigma_{t,g}(\underline{r}) \psi_g^{l+1/2}(\underline{r}, \underline{\Omega}) &= \sigma_{s,g \rightarrow g} \phi_g^l(\underline{r}) + \\
&+ \sum_{g'=1}^{g-1} \sigma_{s,g' \rightarrow g} \phi_{g'}^{l+1/2}(\underline{r}) + Q_g(\underline{r}) \quad (3) \\
g &= 1, \dots, G
\end{aligned}$$

where l is the inner iteration index,

$\psi_g(\underline{r}, \underline{\Omega})$ is the angular flux for group g ,

$\phi_g^l(\underline{r}) = \int \psi_g^l(\underline{r}, \underline{\Omega}) d\underline{\Omega}$ is the transport scalar flux,

$$Q_g(\underline{r}) = \frac{\chi_g}{k_{eff}} \Phi^{CD}(\underline{r}) + \sum_{g'=g+1}^G \sigma_{s,g' \rightarrow g} \phi_{g'}^{CD}(\underline{r}) \quad \text{source from diffusion,}$$

CD refers to the corrected diffusion calculation.

Equation (3) describes the transport inner with the source fixed from a previous “corrected multigroup diffusion” calculation. The corrected diffusion equation has the same form as the conventional diffusion (Eq. (2) above) except that the diffusion parameters have been changed according to the DSA procedure (see page 8-14 of this document). The scalar flux for the next transport iterate, ϕ_g^{l+1} , comes from the solution of the DSA equation for this group, i.e.:

$$-\nabla \cdot \underset{\sim}{D}_g^{l+1/2} \cdot \nabla \phi_g^{l+1} + \sigma_{R,g} \phi_g^{l+1}(\underline{r}) = \sum_{g'=1}^{g-1} \sigma_{s,g' \rightarrow g} \phi_{g'}^{l+1}(\underline{r}) + Q_g(\underline{r}) \quad (4)$$

This is written for the diffusion correction method; see page 8-14 for the definition. In the default transport iteration mode, only one iteration of Eqs. (3) and (4) is done per group until the fission and upscatter sources have converged (this convergence is described in the section: transport source iteration convergence, below). Once each group has processed one inner iteration, the parameters for the corrected, multigroup diffusion equation have been generated as well as the transport scalar fluxes, $\phi_g^{l+1}(\underline{r})$.

These are then used to start another multigroup diffusion iterative solution analogous to Eq. (2) except with the transport corrected diffusion equation (again see Eq. (11) on page 8-16). This process defines the transport outer iteration which continues until the fission and upscatter sources have converged. Once this convergence has been obtained, then IITL is set to IITM and the inner iteration process for each group is continued until the scalar flux has converged (see inner iteration convergence, below), or until IITM is attained, whichever comes first. This usually completes the iteration process of the

transport solution, although, it is possible that converging the inners has caused a change in the fission source that exceeds the convergence criterion. If this is so then another transport outer is performed.

For problems that contain neither fission nor upscatter, the transport outers are not done, IITL is defaulted to IITM, and only the inner iteration DSA convergence procedure is done.

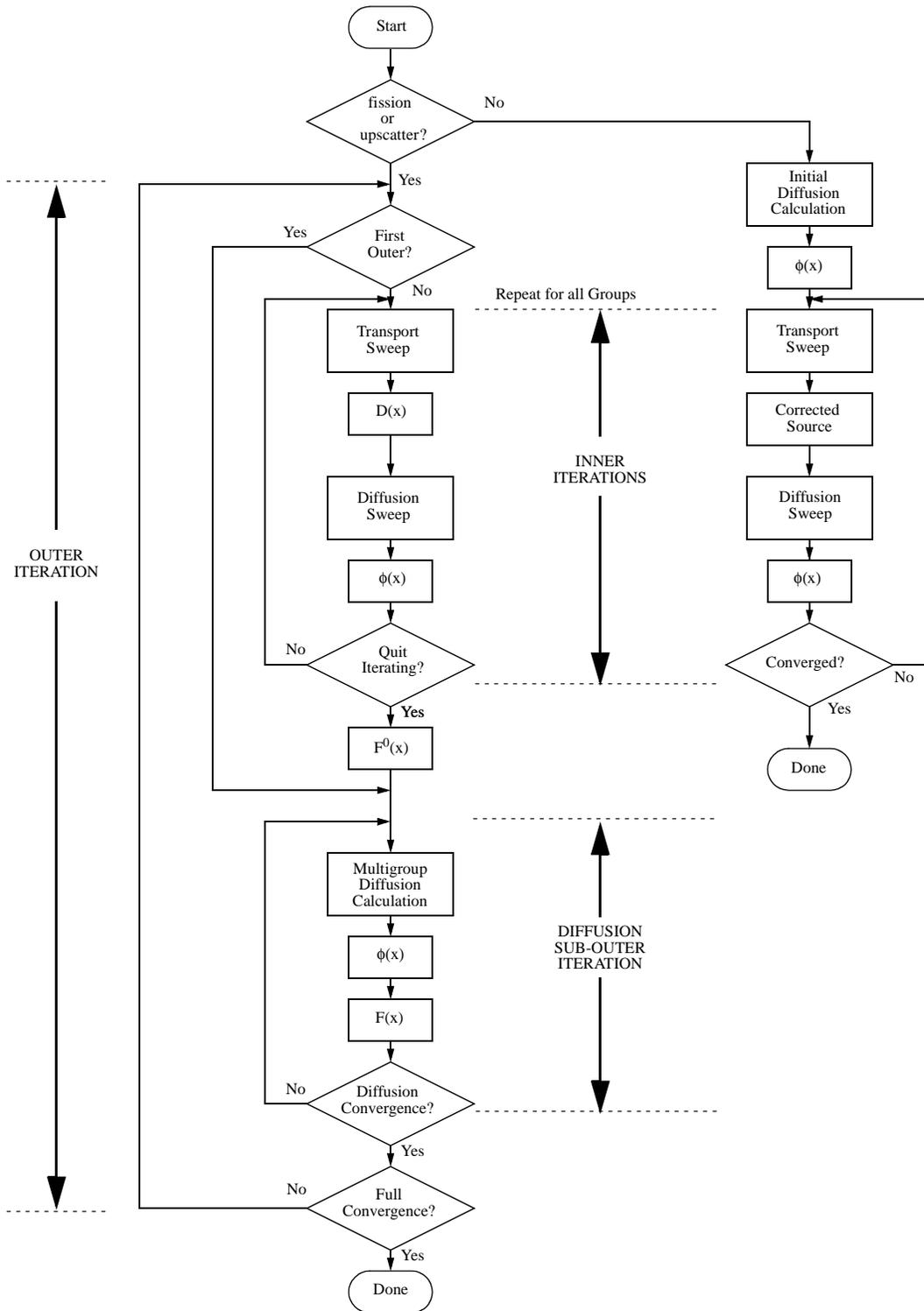


Figure 3.1 Simplified flow diagram of SOLVER iteration strategy.

Convergence Criteria

The convergence of the iterations is monitored at both the inner and the outer iteration level. The input parameters that control the number of iterations are EPSI, EPSO, IITL, IITM, and OITM found in Block-V of the solver module input.

Inner Iteration Convergence.

The inner iterations for a given energy group are said to be converged when the point-wise scalar fluxes from one inner iteration to the next satisfy the condition:

$$\max \left| \frac{(\phi_{i,g}^{l+1} - \phi_{i,g}^l)}{\phi_{i,g}^l} \right| < EPSI , \quad (5)$$

where $\phi_{i,g}^l$ is the scalar flux for mesh point i , group g , and inner iteration l , and where EPSI is the user-input inner iteration convergence criterion.

Diffusion Sub-Outer Iteration Convergence.

The convergence of the diffusion sub-outer iterations requires the satisfaction of two criteria. We use the index k to denote the diffusion sub-outer iteration number, p to denote the transport outer, g to denote the group number, and i the spatial point. Convergence of the diffusion sub-outers is then satisfied when both

$$\max_i \left| \frac{(\Phi_i^{k+1} - \Phi_i^k)}{\Phi_i^k} \right| < 3.0 * EPS \quad (6)$$

and

$$\left| 1 - \lambda_D^{k+1} \right| < EPS . \quad (7)$$

where

Φ^k is the fission distribution at sub-outer k ,

$$EPS = \max(EPSI, 0.01*(0.1**p)) ,$$

$$\lambda_D^{k+1} \equiv \frac{(\Phi^{k+1}, 1)}{(\Phi^k, 1)} .$$

The notation (F,G) denotes the inner product, or volume integral, of the product F*G. In the above, λ_D^{k+1} , measures the precision of the k eigenvalue. Note that the definition of EPS involves a strategy where the diffusion solution convergence depends upon the transport outer iteration, eventually attaining the precision EPSI.

Transport Source Iteration Convergence.

The convergence of the transport source triggers the increase of the number of transport inners from IITL=1 to IITL=IITM if the default strategy is taken. We measure the precision of the transport fission source by comparing the change from the end of the diffusion sub-outers to that resulting from the next transport inners. That is the fission source is deemed sufficiently converged if

$$\max_i \left| \frac{(\Phi_i^{P^*} - \Phi_i^P)}{\Phi_i^P} \right| < 10.0 * EPSI \quad (8)$$

and

$$\left| 1 - \lambda^{P^*} \right| < 1.5 * EPSI , \quad (9)$$

where p* denotes the fission distribution evaluated from the transport scalar flux resulting from the transport inners after the completion of transport outer p (which is the completion of the solution of the DSA multigroup diffusion equation). Thus,

$$\lambda^{P^*} \equiv \frac{(\Phi^{P^*}, 1)}{(\Phi^P, 1)} . \quad (10)$$

Final Convergence

Each of the iterative loops (inner iterations, diffusion sub-outer iterations, and outer iterations) is terminated when either the convergence criteria for that loop are met or when a specified maximum number of iterations have been attained.

For inner iterations the number of iterations is limited by the user input parameter IITL. If the user elects to omit this quantity, the code chooses an appropriate default value.

In problems where outer iterations are not required, that is, fixed-source problems with no upscatter and no fission, the value of IITL is usually chosen to be large, say 20-50, in order that the pointwise fluxes be allowed to meet the convergence criterion before the number of inner iterations reaches IITL.

For eigenvalue problems ($IEVT > 0$), the default procedure is to allow only one inner iteration per group until the fissions, upscatter sources, and diffusion scalar fluxes have neared full convergence. When this is achieved, the allowable number of inner iterations is increased to IITM (a user input quantity) which typically is in the range of 20-50 in order to permit full convergence of the transport fluxes. The assumption here is that it is most efficient to first converge the fission/upscatter sources and then to converge the pointwise fluxes. Final convergence is obtained when the change in the fission distribution and the eigenvalue is less than EPSI, i.e., when

$$\left| 1 - \lambda^{P+1} \right| < EPSI ,$$

and

$$\max_i \left| \frac{(\Phi_i^{P+1} - \Phi_i^P)}{\Phi_i^P} \right| < EPSI \quad (11)$$

These outer iterations are thus terminated when either the above full convergence criteria are met or when the number of outer iterations reaches OITM, a user-input quantity. If not supplied by the user, the code will default the value of OITM to 20.

Iteration Monitor Print

In the printed output from the solver module, an iteration monitor print is supplied for the user. The user should always inspect this monitor print to determine whether or not the problem has successfully converged.

General Aspects of the Monitor Print.

At the end of each outer iteration the monitor provides the elapsed computer time in seconds, the outer iteration number, and the number of diffusion sub-outer iterations required. A number of sub-outer iterations of 100 implies that the diffusion sub-outer iteration did not converge to the criteria of Eqs. (6) and (7) before reaching the maximum allowable number of sub-outer iterations. Also provided is a message as to whether or not the inner iterations satisfied their convergence criterion, Eq. (5). Finally

are included the values of $\lambda^P - 1$ and the maximum pointwise flux error corresponding to the values used in the test for full convergence given by Eq. (11).

In addition to the basic outer iteration information described above, the monitor print provides an inner iteration monitor for certain outer iterations. This inner iteration monitor is normally provided for a fixed-source problem without fission or upscatter (IEVT=0) since only one outer iteration is required. The inner iteration monitor print is triggered by IITL being equal to IITM; this condition is assured by the defaults for these quantities when IEVT=0. The user may thus suppress this portion of the print by setting IITL to some number different from IITM. For other problems (IEVT \neq 0) the inner iteration monitor is only provided for outer iterations following the satisfaction of the “nearly converged” conditions of Eqs. (8) and (9) at which time IITL is set equal to IITM by the code. In the inner iteration monitor are included the group number; the number of inner iterations taken, the maximum pointwise scalar flux error (see Eq. (11)), and the spatial mesh point where this maximum error occurred.

Warning Messages and Their Meanings

In the inner iteration monitor several warning messages are provided for the user if the calculation encountered some difficulty.

A message “ACCELERATION DISABLED” is printed when the transport correction to the diffusion coefficient, diffusion source, or diffusion removal term is such that the synthetic diffusion equation cannot be applied to accelerate that inner iteration. The presence of the message does not necessarily make the answers suspect if convergence is achieved; it merely tells the user that the inner iteration could not be accelerated.

The message “TRANSPORT FLUXES BAD” is a more serious warning. It is provided when nonpositive transport scalar fluxes exist following the last inner iteration. The presence of nonpositive scalar fluxes causes the diffusion inner iteration acceleration to be disabled. Although such a condition is not fatal, it does usually indicate that the spatial mesh is too coarse and that the results are suspect.

Boundary Conditions

Several boundary condition options are available to the user as follows:

- Vacuum boundary condition -- the angular flux on the boundary is identically zero for all incoming directions
- Reflective boundary condition -- the incoming angular flux on the boundary is set equal to the outgoing angular flux in the direction corresponding to specular reflection.
- Periodic boundary condition -- the incoming angular flux on one boundary is set equal to the outgoing angular flux in the same direction on the opposite boundary.

- White boundary condition -- the incoming angular fluxes on the boundary are each set equal to the single value chosen such that the net flow across the boundary is zero, that is, in the one dimensional case, for example,

$$\Psi_{\text{incoming}}(m) = \frac{\sum w_n \mu_n \Psi(\mu_n)_{\text{outgoing}}}{\sum w_n \mu_n},$$

where the sums range over all outgoing directions. This condition is used primarily for cell calculations in cylindrical and spherical geometries where it is applied to the right (outer radial) boundary.

The above boundary conditions are controlled by the Block-V input parameters, IBL (left boundary), and IBR (right boundary). For planar geometries (IGEOM=1), both IBL and IBR must be specified. For curvilinear geometries (IGEOM=2 or 3), only IBR need be specified since the left boundary is assumed by the code to be at the radial origin (r=0), for which the curvilinear geometry, r=0 boundary condition is the only physical condition possible.

Note: Use of a reflective boundary condition (IBL or IBR=1) requires the S_n quadrature set to be symmetric about $\mu = 0$.

Additional input parameters IBT and IBB are used for the top and bottom boundaries of two-dimensional problems and IBFRNT and IBBACK are used for the front and back of three-dimensional problems.

Input of Quadrature Sets

The PARTISN code package has the option of obtaining the discrete-ordinates angular quadrature coefficients from a SNCONS standard interface file¹, from one of three built-in sets in subroutine SNCON, or from card-image input. The input parameter IQUAD in Block-I of the card-image input specifies the source of these coefficients. The number of quadrature coefficients, MM, is determined from the input S_n order parameter ISN and the geometry specification input parameter IGEOM, both found in input Block-I. Values of MM are shown in Table 8.4, “Number of Quadrature Points, M as a Function of S_n Order, N” on page 8-29.

The built-in constants provided in the solver module are

- (i) the Gaussian P_N constants (IQUAD=1) for $S_2, S_4, S_6, S_8, S_{12}, S_{16}, S_{20}, S_{24}, S_{32}$, and S_{48} ;

(ii) the double Gaussian DP_N constants (IQUAD=2) for $S_4, S_8, S_{12}, S_{16}, S_{24}, S_{32}, S_{40}, S_{48}, S_{64},$ and S_{96} ; and

(iii) generalized quadrature, GQ_N , constants (IQUAD=4) for $S_2, S_4, S_6, S_8, S_{12},$ and S_{16} .

For most problems the P_N set is satisfactory. For thin-slab problems in which the angular representation for the leakage flux is important, the DP_N set is recommended. For cylindrical or two-angle plane calculations with anisotropic scattering, the GQ_N set is recommended. The GQ_N set for cylinders and two-angle planes is a generalized even-moment fully symmetric quadrature set.

For problems with anisotropic scattering, it is important that the S_n order be chosen sufficiently large such that the spherical harmonic polynomials described in “Spherical Harmonics Expansion of the Scattering Source” on page 8-23 are correctly integrated. Otherwise, the numerical quadrature error may introduce nonphysical contributions to the neutron balance, preventing convergence of the problem to the desired precision.

For user card-image input of S_n quadrature sets through the WGT and MU arrays in Block-V, it is necessary that the sets be correctly ordered as illustrated in Figure 8.4, Figure 8.5, and Figure 8.6. In addition, if the sums $1 - \sum_m w_m, \sum_m \mu_m,$ and $\sum_m w_m \mu_m$

exceed 10^{-5} , an error message is printed. It should be noted that if the user provides the card-image input arrays WGT and MU, the code will use these arrays for the quadrature constants regardless of the value of IQUAD entered in the input, that is, the WGT and MU input arrays will override any other source of quadrature constants.

For the two and three dimensional modules, the default quadrature set is the so called TWOTRAN set which goes up to S_{16} in even orders. This is a triangular set which is based on Gaussian levels measured from the y axis in xy geometry or the z axis in rz and the three dimensional cases. An attempt is made to make this set as symmetric as possible to rotations about the coordinate axes. There is also available a Gauss-Chebyshev set triggered by the Block-V input variable IQUAD =|2|. This set exists up to S_{50} in even values of S_n order and up to S_{100} in steps of 10 from S_{50} . The main attribute of this set is that it exactly integrates the spherical harmonics up to order S_n-1 . For more information, see Ref. 3.

Zone-Dependent Fission Fractions (the CHI Array)

For problems containing fissile or fissionable nuclides it is necessary to provide the code with the groupwise fission fractions, $\chi_g, g=1,2 \dots, NGROUP$. From certain of the cross-section libraries (ISOTXS, GRUPXS, MENDF, etc.) a single χ vector characteriz-

ing the dominant fissionable isotope (U235, U238, Pu239, etc.) can be extracted automatically from the library and used for the problem. Alternatively, a single χ vector can be provided by the user in Block-III through the card-image CHIVEC input array. Again, this is a single χ vector characterizing the dominant fissionable isotope in the problem.

In Block-V of the input is provided the card-image CHI input array that can optionally be used to provide multiple χ vectors so that each zone in the problem can use the fission fraction characteristic of the dominant fissionable isotope in that zone. The availability of such zone-dependent χ 's is particularly useful for problems in which one zone may have a different dominant fissile isotope than another zone. The CHI array is input as N distinct χ vectors ($1 \leq N \leq \text{NZONE}$) with each vector containing NGROUP χ values. The first χ vector is applied to the first zone, where the first zone is that defined by the first entry in the ASSIGN array of Block-IV; the second χ vector is applied to zone number 2 (defined by the second entry in the ASSIGN array), etc. If the total number of χ vectors in the CHI array is less than NZONE (the total number of zones), then the last χ vector in the CHI array is used for all remaining but unspecified zones.

Note that if the CHI array is used in Block-V, its entries will override the χ 's provided either from the cross-section library or from the CHIVEC array in Block-III.

Input of Inhomogeneous Sources

The PARTISN solver module will solve the inhomogeneous transport equation in the multigroup, discrete-ordinates approximation form, using the method outlined in “Iteration Strategy” on page 3-17. The user specifies this type of calculation by setting the input control word IEVT to 0.

The user must supply the specifications for the inhomogeneous sources either in the input or from a FIXSRC³ standard interface file. The inhomogeneous sources may be spatially distributed on the interior of the problem (distributed source) and/or may be external boundary (surface) sources. If the sources are to be input via FIXSRC standard interface file, the user sets the input control word INSORS to 1. If INSORS is not input with value of unity, the user must supply the source specifications in the input of Block-V as described below.

Distributed Source Input

As described in “Spherical Harmonics Expansion of the Inhomogeneous Source” on page 8-28, the inhomogeneous distributed source must be represented by the spherical harmonic expansion in multigroup form (from Eq. (20) on page 8-28) :

$$Q_g(r, \underline{\Omega}) = \sum_{n=1}^{NMQ} (2L+1)R_n(\underline{\Omega})\tilde{Q}_{n,g}(r) \quad , \quad g = 1, \dots, NGROUP \quad (12)$$

Through the SOURCF or the SOURCE and/or SOURCX input arrays in Block-V of the input, the user inputs the $\tilde{Q}_{n,g}(r)$ of Eq. (12). If input is via the SOURCF array, the input values are used directly as $\tilde{Q}_{n,g}(r)$. If input is via either SOURCE or SOURCX (or both) arrays, the input must be supplied such that SOURCE (g,n)* SOURCX (r, n) = $\tilde{Q}_{n,g}(r)$. There are corresponding source distribution vectors in the y and z directions named SOURCEY and SOURCEZ, respectively. The number of moments, NMQ, in Eq. (12) is determined solely from the number of moments supplied in the input arrays. The proper number of moments for a given Legendre order of anisotropy of the distributed source is shown in Table 8.2, “Number of Spherical Harmonics, N, as a Function of Order” for each geometry. For example, if one wishes to enter a P₃ inhomogeneous source in cylindrical geometry, Table 8.2 shows that six spherical harmonics are required for P₃ in cylindrical geometry. Table 8.3 shows that source moments for

the spherical harmonics $P_0(\xi)$, $P_1^1(\xi)\cos\phi$, $P_2(\xi)$, $\frac{\sqrt{3}}{6}P_2^2(\xi)\cos 2\phi$,

$\frac{\sqrt{6}}{6}P_3^1(\xi)\cos\phi$, and $\frac{\sqrt{10}}{60}P_3^3(\xi)\cos 3\phi$ are needed. These moments are defined by Eq. (19a) on page 8-28 using multigroup notation and recalling that for cylindrical geometry μ is replaced by ξ . The six moments to be supplied in the input are thus:

$$\tilde{Q}_{1,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi P_0(\xi) Q_g(r, \xi, \phi) = Q_{0,g}(r) , \quad (13 a)$$

$$\tilde{Q}_{2,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi P_1^1(\xi)\cos\phi Q_g(r, \xi, \phi) = Q_{c,1,g}^1(r) , \quad (13 b)$$

$$\tilde{Q}_{3,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi P_2(\xi) Q_g(r, \xi, \phi) = Q_{2,g}(r) , \quad (13 c)$$

$$Q_{4,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi \frac{\sqrt{3}}{6} P_2^2(\xi)\cos 2\phi Q_g(r, \xi, \phi) = Q_{c,2,g}^2(r) , \quad (13 d)$$

$$\tilde{Q}_{5,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi \frac{\sqrt{6}}{6} P_3^1(\xi)\cos\phi Q_g(r, \xi, \phi) = Q_{c,3,g}^1(r) , \quad (13 e)$$

and

$$\tilde{Q}_{6,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi \frac{\sqrt{10}}{60} P_3^3(\xi)\cos 3\phi Q_g(r, \xi, \phi) = Q_{c,3,g}^3(r) , \quad (13 f)$$

for $n=1, \dots, \text{NGROUP}$.

It should be recognized that the source moments above are not input as continuous variables in space, r , but are input by fine spatial mesh interval i , $i=1, \dots, \text{IT}$.

It is worth noting that most inhomogeneous distributed sources are assumed to be isotropic, so that NMQ in Eq. (12) is unity and the only source moment entered is the zeroth moment.

$$\tilde{Q}_{1,g}(r) = \frac{1}{4\pi} \int_{-1}^1 d\xi \int_0^{2\pi} d\phi Q_g(r, \xi, \phi) = Q_{0,g}(r) , \quad (14)$$

which, in fact, is simply the scalar source distribution.

The units on the input source moments $\tilde{Q}_{n,g}(r)$ are (particles per unit time and unit volume).

Surface (Boundary) Source Input

With a surface (boundary) source present, the incoming angular flux on the surface is set equal to a user-supplied source, Q_m :

$$\Psi(\mu_m)_{incoming} = Q_m .$$

The units on the surface source are the same as those for angular flux.

The user-supplied source is group-dependent and may either be angularly isotropic or angle-dependent. The user-supplied sources may be input either by Block-V card-image input or via a FIXSRC¹ standard interface file.

For card-image input the left boundary surface sources are input via the SILEFT array (angularly isotropic) or the SALEFT array (for angle-dependent sources). Similarly, right boundary surface sources are input via the SIRITE or SARITE arrays. Note that surface sources may only be input at external boundaries of the physical problem and not at internal interfaces. For the angle-dependent surface sources, only the incoming directions are required, and they must be ordered as described below. There are corresponding arrays for the top and bottom boundaries of two-dimensional problems and for the front and back of three-dimensional problems.

For input of surface sources via a FIXSRC standard interface file, the user-input parameter INSORS in Block-V must be set to unity and the appropriate FIXSRC file must be available at the time of code execution.

Angle-dependent surface sources can be input through the SARITE and SALEFT input arrays. Currently, SALEFT can only be used in plane (X, XY, XYZ) geometry. The surface sources are actually angular fluxes, $\Psi_{m,g}$, at the right or left surfaces [m denotes the order index (with m correlated to specific quadrature directions), g denotes energy group]. There are corresponding arrays for the two and three dimensional geometries (SABOTT, SATOP, SAFRNT, SABACK). The order of SARITE and SALEFT entries is as follows:

$$\Psi_{1,1}, \Psi_{2,1}, \Psi_{3,1}, \dots; \Psi_{1,2}, \Psi_{2,2}, \Psi_{3,2}, \dots; \dots$$

- Slab Geometry (IGEOM= 1 or SLAB or PLANE):

The ordering of the angles is as shown below (S_6 used for illustration).

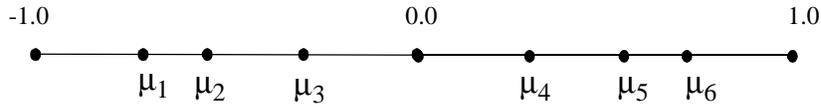


Figure 3.2 Ordering in slab geometry.

with $|\mu_1| = \mu_6$, $|\mu_2| = \mu_5$, and $|\mu_3| = \mu_4$.

Using the S_6 quadrature illustrated above, the correspondence between the SARITE and SALEFT ordering index, m , and the angular directions (shown above) is as follows:

Table 3.1 Source Ordering Index in Slab Geometry.

m	SARITE μ	SALEFT μ
1	μ_1	μ_4
2	μ_2	μ_5
3	μ_3	μ_6

NOTE: The SALEFT ordering differs from the SARITE ordering in terms of the $|\mu|$ associated with the ordered angular fluxes. For example, the first entry in SARITE (for each group) corresponds to μ_1 while the first entry in SALEFT (for each group) corresponds to μ_4 and $|\mu_1| \neq \mu_4$.

- Cylindrical Geometry (IGEOM= 2 or CYLINDER or CYL):

Using S_6 for illustrative purposes, the principal octant ($\mu > 0, \eta > 0$) of the unit sphere of angular directions is as shown below.

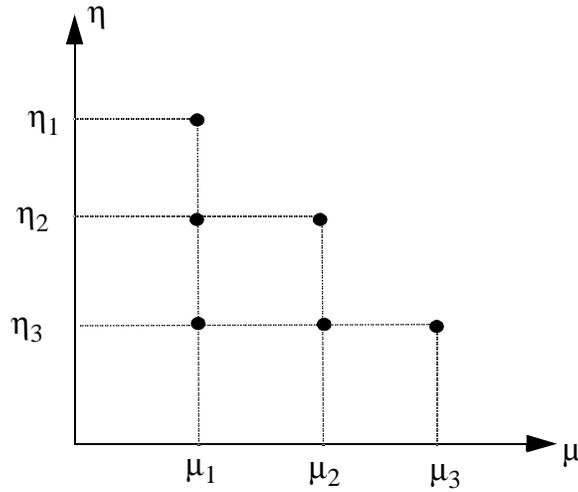


Figure 3.3 Quadrature points in cylindrical geometry.

The order, m , in which the SARITE angular fluxes are entered for each energy group is correlated to the (μ, η) angular directions (shown above) in the table below:

Table 3.2 Source Ordering Index in Cylindrical Geometry

m	μ	η
1	$-\mu_1$	η_1
2	$-\mu_2$	η_2
3	$-\mu_1$	η_2
4	$-\mu_3$	η_3
5	$-\mu_2$	η_3
6	$-\mu_1$	η_3

There are corresponding angular flux arrays for the top and bottom boundaries of two-dimensional problems and for the front and back of three-dimensional problems.

Normalization of the Calculation (the NORM Parameter)

In Block-V of the input is the parameter NORM. Through the use of NORM the user may specify whether or not the calculation should be normalized to a particular particle production rate as described below.

If NORM= 0 or omitted from Block-V no normalization is done.

If NORM= XX, where XX is positive, and inhomogeneous (fixed) sources exist (IEVT= 0), then the calculation will be conducted and normalized such that the total rate at which particles are introduced into the system from inhomogeneous sources is XX. Thus, all fluxes that are in the calculation are normalized to the values they would have if the total rate at which particles are introduced in the system from inhomogeneous sources (distributed and/or surface sources) is XX particles/unit time (or particles if fluxes are to be interpreted as fluences).

If NORM= XX where XX has a positive value, and an eigenvalue calculation is being conducted (IEVT > 0), then the calculation is performed and normalized such that the total rate at which neutrons are produced by fission reactions is XX. Thus, all fluxes that are computed in the calculation are normalized to the values they would have if the total rate at which neutrons are produced by fissions in the system is XX neutrons/unit time (or neutrons if fluxes are to be interpreted as fluences).

Note that NORM applies normalization only to particle production rates. Normalization of edit results to a desired fission power level can be done through the use of the POWER and MEVPER parameters in Block-VI as described in “Power Normalization” on page 2-85. These latter two parameters effect normalization in only the Edit Module.

Transport Corrections for the Cross Sections (TRCOR)

In Block-V of the input is the parameter TRCOR which permits the user to have the solver module perform its calculation using transport-corrected cross sections. Below is provided a little background on the use of the transport correction in PARTISN.

As described in the chapter “PARTISN — METHODS MANUAL”, a truncated spherical harmonics, or Legendre polynomial, expansion of the scattering sources is made per Eq. (14) there. The scattering order ISCT (an input parameter in Block-V) determines where the expansion is to be truncated. Any value of ISCT ($0 \leq \text{ISCT} \leq \text{MAXORD}$) is allowed. Recall that MAXORD is the highest Legendre order for which scattering matrix cross sections exist in the cross-section library being used. The principal reason for choosing to truncate the expansion to an order less than MAXORD is because of computer storage. Each term in the spherical harmonics (Legendre) expansion requires computing and storing an angular flux moment for every energy group and spatial mesh

cell, and the number of such flux moments increases with scattering order (see Table 8.2, “Number of Spherical Harmonics, N, as a Function of Order” on page 8-25).

If ISCT is less than MAXORD, it is usually advantageous to apply a transport correction to the truncated Legendre scattering cross sections. Transport corrections are designed to account approximately for terms in the expansion being omitted by the truncation. There are three different types of transport corrections allowed in the code. They are selected through the use of the parameter TRCOR in Block-V of the input.

- If TRCOR= NO or if TRCOR is omitted, no transport correction is applied.
- If TRCOR= DIAG the diagonal transport correction is applied. With this correction the material macroscopic total cross section for each group and the material macroscopic within-group scattering cross section for each group and each scattering order is modified as shown below:

$$\hat{\sigma}_{t,g} = \sigma_{t,g} - \sigma_s^{ISCT+1}(g \rightarrow g)$$

and

$$\hat{\sigma}_s^l(g \rightarrow g) = \sigma_s^l(g \rightarrow g) - \sigma_s^{ISCT+1}(g \rightarrow g), \quad l = 0, 1, \dots, ISCT .$$

where the notation $\hat{\sigma}$ denotes “transport corrected” cross section. The diagonal transport correction is normally recommended.

- If TRCOR=BHS, the Bell-Hansen-Sandmeier transport correction is applied. With this correction the macroscopic total cross section for each material and group and the macroscopic within-group scattering cross sections for each scattering order, material, and group are modified as follows:

$$\hat{\sigma}_{t,g} = \sigma_{t,g} - \sum_{g'} \sigma_s^{ISCAT+1}(g \rightarrow g')$$

and

$$\hat{\sigma}_s^l(g \rightarrow g) = \sigma_s^l(g \rightarrow g) - \sum_{g'} \sigma_s^{ISCAT+1}(g \rightarrow g'), \quad l = 0, 1, \dots, ISCT .$$

- If TRCOR=CESARO, the nth-Cesàro-mean-of-order 2 transport correction is applied where n=ISCT. With this correction the macroscopic group-to-group scattering cross section for each scattering order, zone, and group is modified as shown below:

$$\hat{\sigma}_s^l(g \rightarrow g') = \frac{(ISCT + 2 - l)(ISCT + 1 - l)}{(ISCT + 2)(ISCT + 1)} \sigma_s^l(g \rightarrow g'), \quad l = 1, \dots, ISCT.$$

This Cesàro correction ensures that the truncated Legendre expansion for the scattering cross section from group g to group g' is (i) positive, (ii) preserves the $l = 0$ (i.e., P_0) moment of the scattering cross section, and (iii) converges to the same value as $\sigma_s^l(g \rightarrow g')$ for large values of ISCT (see Ref. 2).

Note that if ISCT= 0, the Cesàro correction does nothing to the cross sections -- the multiplier of $\sigma_s^l(g \rightarrow g')$ is unity. Generally, the Cesàro transport correction should only be used with ISCT ≥ 2 .

At the time of the writing of this manual, little experience has been had with the Cesàro correction in terms of its accuracy. As a result the user is cautioned regarding its use. However, using this transport correction will eliminate the possibility of negative sources, as discussed in “Warning Messages and Their Meanings” on page 3-26.

Please note that the transport corrections described above are only applied to the macroscopic cross sections used in the solver module. None of the cross-section files are altered by the use of the transport correction in the solver module.

Buckling Corrections

Leakage from the transverse dimension(s) of a multidimensional system may be simulated in a lower dimensional solver by using a user-specified buckling height (BHGT) and/or buckling width (BWTH) in the Block-V card-image input. For plane geometries (IGEOM=1), both BHGT and BWTH may be specified. For cylindrical geometry only the buckling height, BHGT, may be specified. The buckling dimensions are in units consistent with the units on cross section, for example, in cm if macroscopic cross sections are in cm^{-1} . If diffusion theory is assumed adequate, then the flux shape in the transverse directions, say z , is of the form $\cos \pi z / \tilde{h}$ so that the flux shape function vanishes at the extrapolated system half-heights $\pm \tilde{h} / 2$. Applying this to the transport equation the transverse leakage appears as a buckling absorption with a buckling absorption cross section given by

$$\sigma_{a, BHGT} = \frac{\sigma}{3} \left[\frac{\pi}{\sigma * BHGT + 1.4209} \right]^2,$$

where σ is the macroscopic zone total cross section, BHGT (or similarly BWTH) is the buckling height (or buckling width), and $1.4209/\sigma$ is twice the Milne planar extrapolation distance.

The buckling absorption correction is applied to both the total cross section and absorption cross section for each group and zone in the physical problem. Consequently, the absorption rate printed in the output solver module coarse-mesh balance table contains this buckling absorption.

Eigenvalue Searches

It is possible to perform an eigenvalue search on material concentration (concentration search), system dimensions (dimension search), or the time absorption (alpha search) to achieve a desired value of k_{eff} . The type of search is controlled by the input parameter IEVT supplied in Block-V of the card-image input as follows:

<u>IEVT^a</u>	<u>Type of Eigenvalue Search</u>
2	Time absorption (alpha)
3	Concentration
4	Critical size (dimension)

- a. Not included here are the options IEVT=-1 for inhomogeneous source problems with upscatter and/or fission, IEVT=0 for inhomogeneous source problems without upscatter or fissions, and IEVT=1 for k_{eff} calculations.

For time-absorption calculations, the time-dependent angular flux is assumed to be separable in time and space, viz.,

$$\psi(r, \underline{\Omega}, t) = \psi(r, \underline{\Omega})e^{\alpha t}$$

If this assumption is inserted into the time-dependent transport equation, the exponentials cancel and a fictitious cross-section term of the form α/v_g appears as a correction to the total and absorption cross sections. Here v_g is the neutron speed associated with energy group g . The exponential factor α is then the eigenvalue sought in the time-absorption eigenvalue search. Obviously, $\alpha = 0$ for an exactly critical system, and $\alpha > 0$ for a supercritical system.

For concentration searches, the material concentrations are modified in accordance with the description provided under the ASGMOD array in Block- IV of the card-image input (See “Mixing Array for a Concentration Search” on page 2-53).

For dimension searches, the coarse-mesh boundaries can be modified selectively to obtain a critical system. The modified coarse-mesh boundaries, \tilde{R}_k are calculated from the initial input boundaries, R_k

$$\tilde{R}_{k+1} = \tilde{R}_k + (R_{k+1} - R_k) * (1 + EV * RM_k), \quad k=1,2,\dots, IM, \quad (15)$$

where EV is the eigenvalue sought in the search. The factors RM_k are the coarse-mesh radii modifiers which are input by the user via the RM array in the Block-V card-image input, and control how the coarse-mesh boundaries are modified. Clearly, if RM_k is zero, the thickness of the k^{th} zone is not altered. If all RM_k are unity, the system dimensions are uniformly expanded ($EV < 0$) or contracted ($EV > 0$). Many sophisticated changes can be made, limited only by the ingenuity of the user. For example, an interface between two zones may be moved while the remainder of the system is left unchanged.

In all three types of searches the appropriate system parameter may be adjusted to achieve the desired value of k_{eff} . This value is taken to be unity (criticality) unless the input parametric value type (IPVT in Block-V of the card-image input) is set to unity. If IPVT=1, the desired parametric value of k_{eff} is input by the user as PV (in Block-V).

For concentration searches (IEVT=3) and dimension searches (IEVT=4), it is also possible to adjust the appropriate system parameter to achieve a system whose neutral particle flux is changing exponentially in time at the rate $e^{\alpha t}$ by setting the input parametric value type, IPVT, to 2. In this case the user enters the desired exponential factor α as the parametric value PV in the input. Note that an α of 0.0 corresponds to a normal concentration or dimension search on a k_{eff} of unity.

It is important to recognize that the value of PV input by the user remains fixed throughout the search process.

Regardless of the parameter being adjusted, the search is executed by performing a sequence of k_{eff} type calculations, each sequence for a different value of the parameter being treated as the eigenvalue. The search is for a value of the parameter that makes the value of λ unity where λ is defined as

$$\lambda = \frac{(\text{Fission source})^k + \text{Inhomogeneous source}}{(\text{Fission source})^{k-1} + \text{Inhomogeneous source}} \quad (16)$$

for the k^{th} outer iteration. The search is controlled by the subroutine NEWPAR in the solver module.

In the following description of NEWPAR, it is helpful to refer to Figure 3.4 in which the deviation of λ from unity for each outer iteration is plotted.

For the initial system, NEWPAR continues the outer iteration until two successive values of λ differ by less than EPSO. For subsequent sequences of λ values, a different convergence precision, XLAX, is used. After the first converged λ sequence is obtained, the initial value of the eigenvalue (EV) is altered by EVM, an input value. If $\lambda > 1$ (multiplying system), the new eigenvalue is equal to EV+EVM; if $\lambda < 1$ (decaying system), the new value is equal to EV-EVM. Thus, the sign and value of EVM should be chosen such that the use of EV+EVM will reduce the reactivity of the system. Conversely, the use of EV-EVM should increase the reactivity of the system.

Basically, after two converged values of λ are obtained for two different system configurations, subroutine NEWPAR attempts to fit a curve through the most recent values to extrapolate or interpolate to a value of unity. Depending on the amount of information available and the size of $|1 - \lambda|$, this fit proceeds in different ways. A parabolic fit cannot be made until three converged values of λ are available, and is not attempted unless $|1 - \lambda|$ is greater than an input search lower limit (XLAL) and less than an input search upper limit (XLAH). If a parabolic fit is tried and the roots are imaginary, a straight-line fit is used. If the roots are not imaginary, the closest root is used as the new value of EV. Once a bracket is obtained (change of sign of $\lambda - 1$), the fit procedure is not allowed to move outside the region of the bracket. Should a parabolic fit select an eigenvalue outside the bracket region, this value is rejected and the new value is taken to be one-half the sum of the previous value and the value previous to that.

Whenever the parabolic fit is not used, a linear fit is used and the new eigenvalue is computed from

$$(EV)_{\text{new}} = (EV)_{\text{old}} + \text{POD} * \text{EVS} * (1 - \lambda) , \quad (17)$$

where POD is an input “parameter oscillation damper” that may be used to restrict the amount of change in the eigenvalue. In Eq. (17), EVS is a measure of the slope of the curve. When $|1 - \lambda| > \text{XLAH}$, $(1 - \lambda)$ in Eq. (17) is replaced by XLAH (with the correct sign) to prevent too large a change in EV. After $|1 - \lambda| < \text{XLAL}$, the value of EVS is fixed and kept constant until convergence to prevent numerical difficulty in the approximation of the derivative when λ is close to unity.

Because parametric search problems represent sequences of k_{eff} calculations, it behooves the user to study the use of subroutine NEWPAR in order to optimize his calculations. It also behooves the user to pose soluble problems. That is, there are many

problems, especially concentration searches, for which solutions are not possible, and discovering this by trial and error is the hard way. Ideally, the user will have some estimate of the critical parameter available from a lower order computation.

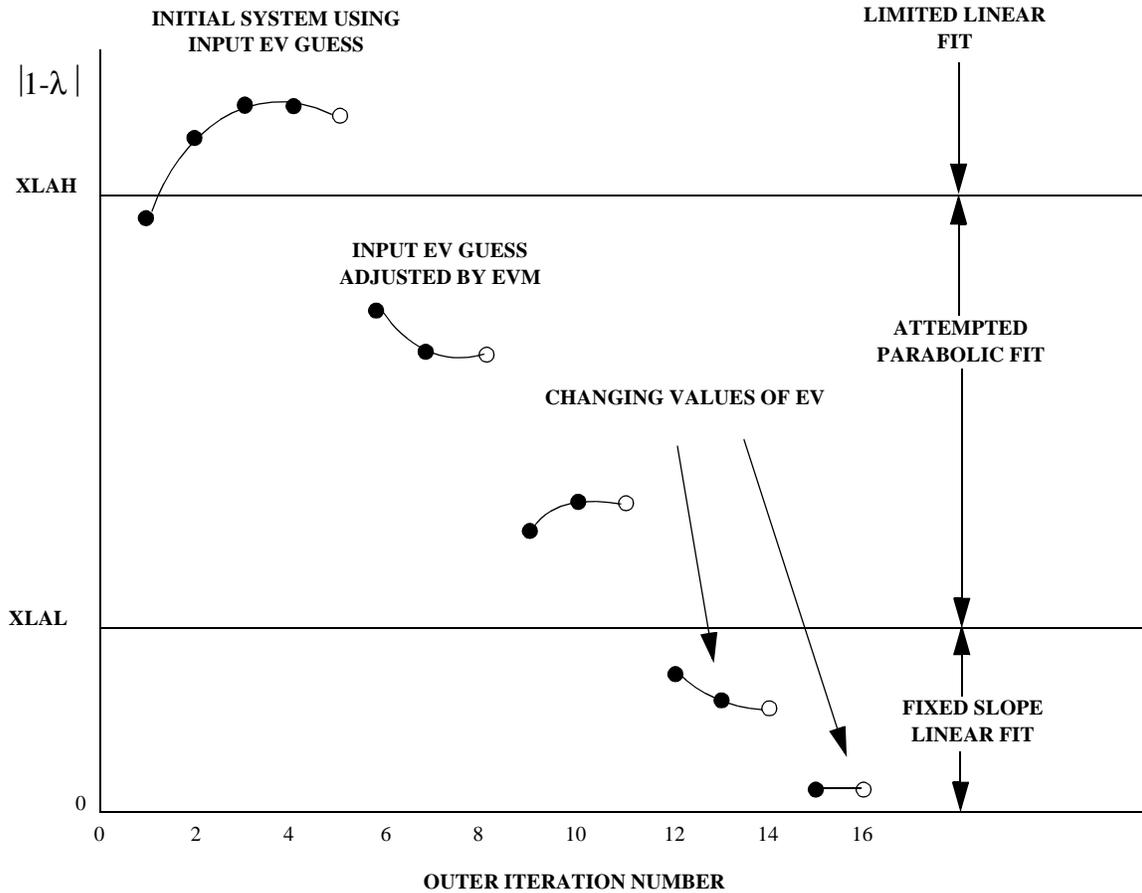


Figure 3.4 Variation of λ during a hypothetical eigenvalue search.

Convergence in time-absorption calculations is typically one-sided. If EV (the eigenvalue α) is negative, then there is a possibility that the corrected removal cross section will become negative. If this happens, the automatic search procedure may fail dramatically. For this reason $POD=0.5$ or less is frequently used in such searches.

Adjoint Computations

The PARTISN solver module solves the adjoint transport equation by transposing (in energy) the matrices of scattering cross sections and inverting the group order of the problem. The transposition of the scattering matrix converts a downscatter problem to an upscattering problem so that by inverting the group order the problem will execute in

a downscatter-like mode. In addition to transposing the scattering matrices, the fission source term in the transport equation is transposed so that instead of $\chi_g \Sigma(v\sigma_f)_h \phi_h$, one has $(v\sigma_f)_g \Sigma \chi_h \phi_h$. The code does not transpose the angular direction matrix associated with the leakage terms in the transport equation. Instead, the adjoint calculation of the leakage operator proceeds as in the direct (forward) calculation, but the results of the adjoint calculation for direction Ω must be identified as the adjoint solution for direction $\bar{\Omega}$. For example, the vacuum boundary condition at a surface (no incoming angular flux) in an adjoint calculation must be interpreted as a condition of no outgoing flux. Likewise, the adjoint leakage at a surface must be interpreted as incoming instead of outgoing.

All group-order inversions and fission source and scattering matrix transpositions are performed by the code; the user need only set the input parameter ITH in Block-V to unity to effect an adjoint calculation. (If the problem contains inhomogeneous sources, these sources must quantitatively be, of course, the adjoint sources.)

The printed output from the solver module in an adjoint calculation indicates the correct group ordering and need not be inverted by the user. The adjoint fluxes from the solver module are written to a binary ATFLUX¹ standard interface file.

In performing the adjoint reversals of the scattering matrices and the group inversions, the code prepares a binary, code-dependent interface file ADJMAC. This ADJMAC file contains the adjoint-reversed material cross sections to be used by the solver module. ADJMAC is essentially the adjoint-reversed counterpart to the MACRXS file described in “INTERFACE FILES USED IN MIXING” on page 7-17, and the rules for saving and using an existing ADJMAC file are the same as for an existing MACRXS file.

The performance of adjoint edits is described in “Adjoint Edits” on page 4-15.

REFERENCES

1. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
2. T. E. Albert and P. Nelson, "Computation of Azimuthally Dependent Albedo Data by Invariant Embedding," in Proc. of Sixth Intl. Conf. on Radiation Shielding, May 16-20, 1983, Tokyo, Vol. I, pp. 283-293.
3. R. D. O'Dell and R. E. Alcouffe, "Transport Calculations for Nuclear Analysis: Theory and Guidelines for Effective Use of Transport Codes," Los Alamos National Laboratory report LA-10983-MS (September 1987).

INPUT DETAILS CHAPTER INDEX

A

Acceleration	
Upscatter(Grey)	3-25
ACCELERATION DISABLED message	3-26
Adjoint	
Calculation of	3-43
Group order in	3-43
Albedoes	
Use of	3-28
Alpha(time absorption) search	3-39
Approximation	
Transverse leakage	3-38

B

Boundary condition	
Discussion of	3-27
Boundary source	3-33
Buckling	
Simulated leakage	3-38

C

Chi	
Zone dependent	3-30
Coarse mesh	
Definition of	3-15
Convergence	
Behavior in search	3-42
Iterating to	3-17
Tests	3-23
Cross section library	
Transport correcting	3-36

E

Energy groups	
Order in adjoint run	3-43
Extraneous source, see Source, inhomogeneous	

F

Fine mesh	
Definition of	3-15
Fission fraction	
Zone dependent	3-30

Flux
 Normalization of 3-36

G

Geometry
 Coarse mesh 3-15
 Fine mesh 3-15
 Grey acceleration 3-25

I

Inner iteration convergence 3-23
 Iteration
 Monitor print of 3-25
 Monitor, warning message from 3-26
 Strategy of 3-17

M

Memory requirements 3-12
 Message
 Warning from iteration monitor 3-26
 Monitor
 Iteration, print of 3-25

N

NEG. SOURCE - ACCELERATION DISABLED message 3-27
 NEG. SOURCE - TRANSPORT FLUXES BAD message 3-27
 Normalization
 Of the flux 3-36
 Of the source rate 3-36

Q

Quadrature sets
 ONEDANT, choices of 3-28

S

Searches
 Convergence behavior 3-42
 General control of 3-39
 Time absorption(alpha) 3-39
 Source
 Boundary angular fluxes 3-33
 Inhomogeneous, input of 3-31
 Normalization of 3-36
 Storage
 Memory requirements 3-12

T

Terminating run 3-24
 Time absorption(alpha) search 3-39
 Transport correction of cross section library 3-36
 TRANSPORT FLUXES BAD message 3-26

U

Upscatter acceleration 3-25

V

Void, specifying a 3-16
 Volumetric source, input of 3-31

W

Warning messages
 From iteration monitor 3-26

Z

Zone
 Concept/Definition of 3-16

Specifying a void in 3-16

RUNNING THE EDIT MODULE

Transport Methods Group, CCS-4
Los Alamos National Laboratory

4

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	4-3
LIST OF TABLES.....	4-5
INTRODUCTION	4-7
REACTION RATES.....	4-9
Spatial Options for Edits	4-9
Energy Group Options For Edits.....	4-10
Forms Of Response Functions	4-11
Cross-Section Response Functions: EDXS	4-11
User-Input Response Functions: RSFE, RSFX	4-13
Response Function Summing Options	4-13
Cross-Section Response Functions Sums: MICSUM	4-13
User-Input Response Functions Sums: IRSUMS	4-14
Adjoint Edits	4-15
ASCII File Output Capabilities (the EDOUTF Parameter)	4-15
MASS INVENTORIES	4-17
EDIT CHAPTER INDEX.....	4-19

LIST OF TABLES

Table 4.1: MASSED Input Values	4-17
Table 4.2: MASSED Input Values for Fine Mesh Mix Problem.....	4-18



INTRODUCTION

The basic function of the Edit Module is to perform postprocessing, or edit, operations after the flux calculation is done. Two basically different types of edits may be done: a simple inventory of the masses in the problem, and a calculation of reaction rates using the multigroup, pointwise scalar fluxes generated by the execution of the solver module or, perhaps, by some other neutronics code. The Edit Module uses the scalar fluxes, multiplies them by suitable quantities hereafter called response functions, calculates sums of these products over space and/or energy (if desired), and produces printed output of the results.

The Edit Module is essentially a free-standing code module accepting only interface or files as input. Most of these interface files are general in nature in that they apply both to the Solver and the Edit Modules (see Table 9.1, “Files Read and Written,” on page 9-10). Included in these general files are the geometry specifications (GEODST or LNK3DNT file), the material mixing and cross-section specifications (NDXSRE, ZNATDN, and SNEXDTE files), and the assignment of materials to zones specifications (ASGMAT file). Another general file required by the Edit Module is a standard scalar flux interface file, either regular (forward) scalar fluxes (RTFLUX file), or adjoint scalar fluxes (ATFLUX file). Either an RTFLUX or an ATFLUX file is automatically provided by the Solver Module whenever it is executed. The specific edit operations to be performed using the information from the above general files are provided to the Edit Module by means of an EDITIT interface file. This file is created by the Input Module solely from user card-image input in Block-VI of the input data.

Because of the structure and interface file linkage of the PARTISN code, several different Edit Module runs can be performed using the same set of general files. For example, once the Solver Module is executed and its scalar flux interface file written, the Edit Module can be repeatedly executed without re-execution of the Solver Module. Only the Edit Module card-image input need be changed so that a new EDITIT file is created between runs.

The remainder of this chapter provides details pertinent to the editing options available to the user in the Edit Module card-image input (Block-VI).

REACTION RATES

Spatial Options for Edits

Edits can be performed on the fine spatial mesh points (as specified in Block-II geometry input) or on integrals over specified spatial intervals (called Edit Zones).

The fine space-point option is chosen by setting the Block-VI input parameter PTED to unity. In this form the edit quantity, denoted by ρ , for the i^{th} spatial mesh point is computed as

$$\rho_{i, g'} = \sum_{g \in g'} \phi_{i, g} R_{i, g} \quad , \quad (18)$$

where:

$\phi_{i, g}$ = scalar flux for mesh point i , energy group g .

$R_{i, g}$ = a response function which may be either input directly via the RSFE and RSFX arrays (below) or formed from input cross sections.

g' denotes an Edit energy-broad-group (See “Energy Group Options For Edits” on page 4-11) consisting of one or more Solver energy groups.

With the Block-VI input parameter BYVOLP set to unity, the above edit quantity will be multiplied by the mesh interval “volume” V_i . The user may also select those points, or intervals, for which he wishes point edits by use of the POINTS input array in Block-VI. If PTED=1 and the POINTS array is not specified, the code will provide output for all mesh points (default).

In three-dimensional problems a special point edit by planes has been included. Input specifications in Block-VI includes the arrays IPLANE, JPLANE, and KPLANE. These are a set of integers which designate which planes of point edits will be printed. If any of these arrays are specified, then the POINTS array must not be specified. Thus, to get point edits of all the points on the x-y planes, 1, 5, and 10, for example, set KPLANE = 1,5,10.

To interface with a specific graphics package, TECPLOT© which is commercially available, the edit module also has the capability to generate TECPLOT files which mirror the printed output above. This capability is available for 2- and 3-dimensional geometries only. The generation of these files and the printing of the point edits is con-

trolled by the input parameter PRPLTED in Block-VI. Thus, PRPLTED = 0/1/2/3 is chosen to print only / no point edit output / TECPLOT files only / both generate the TECPLOT files and print the point edits.

To obtain edit quantities that are integrals over desired spatial intervals, the input quantity ZNED is set to unity. The desired spatial intervals, called Edit Zones, are specified by the user through the EDZONE array in input Block-VI. In specifying the Edit Zones through the EDZONE array the following rules must be observed:

- (i) each and every fine-mesh interval (point) must be assigned to an Edit Zone, that is, given an Edit Zone number,
- (ii) Edit Zone boundaries are arbitrary, that is, they are independent of coarse-mesh or material boundaries.
- (iii) Edit Zone numbers must be positive integers in the range 1, 2, ..., N where N is the total number of Edit Zones desired.

Example: Given a one-dimensional problem with 30 mesh intervals, it is desired that edit quantities be produced that are integrals over the first 10-mesh intervals, the second 10-mesh intervals, and the remaining 10-mesh intervals. There are thus 3 Edit Zones each comprising 10-mesh intervals. Using the free-field repeat option shown in the chapter "FREE FIELD INPUT REFERENCE," the EDZONE specification could be provided as EDZONE = 10R1, 10R2, 10R3 to specify that the first 10-space intervals are in Edit Zone 1, the second 10 in Edit Zone 2, and the third 10 in Edit Zone 3. It should be noted that the ordering of the Edit Zones 1, 2, and 3 with the first, second, and third set of 10-mesh points is not required.

Thus, with ZNED=1, the EDIT module will produce edit quantities, ρ , for Edit Zone Z_m as

$$\rho_{Z_m, g'} = \sum_{g \in g'} \sum_{i \in Z_m} \phi_{i, g} R_{i, g} V_i \cdot \quad (19)$$

If Edit Zone edits are requested (ZNED=1) and the EDZONE array is not specified, the code will assume a default specification of the Edit Zones equal to the Coarse-Mesh intervals (see XMESH input array in Block-II).

IMPORTANT NOTE: In order to get printed output from the EDIT module, either point edits (PTED=1), or edit zone edits (ZNED=1), or mass edits (MASSED=1), or any combination thereof must be specified.

Energy Group Options For Edits

The user may select the energy-group structure desired for the edit output by means of the ICOLL input array in Block-VI. Through this input array the user can collapse the energy-group structure used in the Solver Module down to fewer (broader) groups for edit purposes.

Example: Consider a 24 energy-group structure used by the Solver Module in which the first 12 groups are considered “fast” groups, groups 13 through 21 are “epithermal” groups, and groups 22 through 24 are “thermal” groups. If it is desired that quantities be calculated as integrals (sums) over the three that edit broad groups denoted fast, epithermal, and thermal, the ICOLL array would be specified as ICOLL=12, 9, 3 to collapse the first 12 groups into Edit Energy-Broad-Group 1 (the “fast” broad group), the next 9 groups (groups 13 through 21) into Edit Energy-Broad-Group 2 (epithermal), and the last 3 groups (groups 22 through 24) into Edit Energy-Broad-Group 3 (thermal).

If the ICOLL array is not specified, the code will assume the default condition of one Solver Module energy group per Edit Energy-Broad-Group.

The IGRPED input parameter in Block-VI is used to control the printed output with respect to the Edit Energy-Broad-Groups. With IGRPED=0 only the energy-group total (sum over all groups) of the edit quantities is printed. With IGRPED= 1 or 2, edit quantities for each of the Edit Energy-Broad-Groups are printed. With IGRPED= 3, edit quantities for each Edit Broad Group plus the energy-group total are printed.

Forms Of Response Functions

As indicated in the preceding sections, reaction rate quantities all involve taking the product of the scalar flux, $\phi_{i,g}$, and a response function, $R_{i,g}$, (for spatial mesh point i and energy group g). In this section are described the various forms that the response function $R_{i,g}$ can take.

Cross-Section Response Functions: EDXS

Response functions can be formed directly from cross-section data. In this case it is necessary to specify the particular type, or types, of cross sections to be used, that is, (n,γ) , (n,α) , total, absorption, etc. The cross-section data provided to the Edit Module on the SNXEDT file will contain a particular cross-section type in a unique position within the cross-section data table as indicated in the table “Edit Cross-Section Types by Position and Name” on page 2-81 or “MENDEF Library Edit Cross Sections” on page 2-88.

Through the EDXS input array in the Block-VI input, the user specifies which cross-section types are desired using either the integer edit position numbers or the character names as given in the same tables.

Example: Consider a problem in which isotope cross sections were supplied by means of an ISOTXS binary file. It is desired that edits be performed using both the n,α and n,γ cross sections. Using the table on page 2-81, the EDXS array would be input as EDXS= "N-ALPH," "N-GAMM" or, alternatively, as EDXS=8, 10.

The specific forms of cross-section-based response functions available in the EDIT module are the resident macroscopic, isotope microscopic, constituent, and material forms. Each of these is described below.

1. Resident Macroscopic Cross-Section Response Functions: RESDNT Input Parameter.

The resident macroscopic cross section, $\Sigma_{i,g}^{RES}$ at mesh point i , energy group g is defined as the actual macroscopic cross section that was used by the Solver Module. To obtain this as the response function, namely

$$R_{i,g} = \Sigma_{i,g}^{RES}$$

the Block-VI input parameter RESDNT is set to unity.

2. Isotope Microscopic Cross-Section Response Functions: EDISOS Input Array. Isotope microscopic cross section, σ_g^{ISO} , may be used for the response functions by identifying the isotopes desired through the EDISOS Block-VI input array. In this edit the cross sections are taken directly from the EDIT module file SNEXDTE, which themselves originally came from the basic cross-section library. Note that the response function

$$R_{i,g} = \sigma_g^{ISO},$$

is spatially constant so that the edit quantity $\sigma_g^{ISO} \phi_{i,g}$ will be calculated at mesh point i even if the isotope was not physically present at that location.

3. Resident Constituent Cross-Section Response Functions: EDCONS Input Array. Resident constituent cross sections, $\Sigma_{i,g}^{ISO}$, can be used for response functions by identifying the isotopes desired through the EDCONS Block-VI input array. The resident constituent or simply, constituent, cross section is a partial macroscopic cross section given by the product of the isotope microscopic cross section times the actual atom density associated with that isotope at the spatial location as seen by the Solver Module. Thus, for a constituent cross-section edit the response function $R_{i,g}$ is

$$R_{i,g} = N_i^{ISO} \sigma_g^{ISO} = \Sigma_{i,g}^{ISO},$$

for spatial mesh interval i , group g .

4. Material Cross-Section Response Functions: EDMATS Input Array.

Material macroscopic cross sections, Σ_g^{MATL} , can be used for the response function by identifying the desired materials through the EDMATS array in the Block-VI input. In this material edit the macroscopic cross sections for the materials specified in the MATLS array in the mixing input block (Block-IV) are reformed using the microscopic cross sections on file SNXEDT together with the mixing instructions stored on the NDXSRF and ZNATDN standard interface files. Thus, for a material cross-section edit the response functions are of the form

$$R_{i,g} = \Sigma_g^{MATL} \phi_{i,g}.$$

Note that these response functions are spatially constant so that the edit quantity $\Sigma_g^{MATL} \phi_{i,g}$ will be calculated at each mesh point, i , even if the material was not physically present at that location.

User-Input Response Functions: RSFE, RSFX

In addition to response functions based on cross-section data, the user may directly input response functions in a space-energy separable form through the Block-VI input arrays $RSFE_g$ and $RSFX_i$ for energy-group g and space-point (interval) i for a one-dimensional case. Thus, for user-input response functions,

$$R_{i,g} = RSFE_g * RSFX_i.$$

NOTE: The RSFE array is required if user-input response functions are desired. The RSFX input array is optional (It is defaulted to unity everywhere). There are RSFY and RSFZ input arrays to specify the distribution in the second and third dimensions, if needed. These arrays are also defaulted to unity everywhere.

The RSFE input array can be used to obtain groupwise fluxes (or sums of groupwise fluxes) by using RSFE array entries of 1.0 in the groups of interest. Fluxes can similarly be renormalized by use of the appropriate normalization factor in either the RSFE or RSFX arrays.

Response Function Summing Options

Certain response summing operations are available to the user by means of the Block-VI input arrays MICSUM and IRSUMS. The MICSUM array provides for the specifica-

tion of cross-section response function summing, while the IRSUMS array provides for the specification of user-input response function summing. Each of these is described below.

Cross-Section Response Functions Sums: MICSUM

Through the use of MICSUM input array in Block-VI, either isotope microscopic edit sums or resident constituent edit sums, but not both, will be computed. (Recall that isotope microscopic edits are invoked by means of the EDISOS input array and resident constituent edits are invoked by means of the EDCONS input array.)

The MICSUM input array is a packed array with data entered as follows: a set of isotope numbers or character names (from the basic isotope input library) is given followed by a set of cross-section type position numbers or character names (See “Edit Cross-Section Types by Position and Name” on page 2-81). These sets are delimited with an entry of 0 (zero). Reaction rates (edit quantities) are calculated for each isotope specified in the set for each cross-section type specified and summed to form the first sum. The next two sets of data are used to define the second sum, etc.

The MICSUM array is only used in conjunction with either the EDCONS array or the EDISOS array as follows:

- If the EDCONS array is specified, the summing defined by the MICSUM array applies to the resident constituent (partial macroscopic) cross sections. Isotopes used in the MICSUM array must have been used in the EDCONS array.
- If the EDCONS array is not specified and the EDISOS array is specified, the summing defined by the MICSUM array applies to the isotope microscopic cross sections. Isotopes used in the MICSUM array must have been used in the EDISOS array.

Example: Suppose the EDCONS array were specified as

EDCONS = PU239, PU240, PU241, U238,

and the MICSUM array were specified as

MICSUM= PU239, PU241, 0, “N-GAMM,” “N-FISS,” 0,
PU240, U238, 0, ABS

For mesh point *i*, energy group *g*, the two sums specified in the MICSUM array would be

$$\text{SUM 1: } \left\{ [N_i(\sigma^{n,\gamma} + \sigma^f)_g]^{PU239} + [N_i(\sigma^{n,\gamma} + \sigma^f)_g]^{PU241} \right\} * \phi_{i,g}$$

$$\text{SUM 2: } [(N_i\sigma_g^a)^{PU240} + (N_i\sigma_g^a)^{U238}] * \phi_{i,g} .$$

User-Input Response Functions Sums: IRSUMS

Through the use of the IRSUMS input array in Block-VI, user-input response function edit sums can be computed. The input to the IRSUMS array is supplied as follows: a set of user-input response function numbers or names is entered and the set is delimited with an entry of 0 (zero). Edit quantities are calculated for each response function specified and the edit quantities summed to form the first sum. The next set of data is used to form the second sum, etc. Only user-input response functions that have been provided through the RSFE input array (and, optionally, the RSFX input array) can be used in the IRSUMS array.

Adjoint Edits

The EDIT module will normally perform regular (forward) edits using regular (forward) scalar fluxes from an RTFLUX standard interface file. If adjoint edits are to be performed, the user need only set the Block-VI input parameter AJED to unity and ensure that the appropriate adjoint scalar flux file (ATFLUX file) exists and is available to the EDIT module at the time of execution. (“Adjoint Computations” on page 3-42 describes adjoint calculations in the SOLVER module.) In the adjoint mode, the EDIT module performs all adjoint reversals and provides the correct group ordering in the printed output.

ASCII File Output Capabilities (the EDOUTF Parameter)

Because of the desirability of presenting output quantities from a calculation in graphics form (plots, tables, etc.), it is valuable to have output in eye-readable ASCII files. Such files are easy to use for extracting information and data and putting it in forms usable by graphics packages, text-editors, etc. The Edit Module can optionally create two such ASCII files for the user. The EDTOUT file is an ASCII file which contains certain geometric information (such as problem geometry, number of coarse mesh intervals, the coarse mesh boundary positions, and number of fine mesh intervals per coarse mesh) together with the edit quantities produced by the Edit Module. The EDTOGX file is an ASCII file containing geometric information, group-total fission source (if problem had fissionable material), and, optionally, the multigroup scalar fluxes at each fine mesh

point. File descriptions for these two files are provided in the chapter “FILE DESCRIPTIONS” starting on page 11-1.

The Edit Module (Block-VI) parameter EDOUTF controls the creation of the EDTOUT and EDTOGX files as described in “ASCII File Output Capabilities (the EDOUTF Parameter)” on page 4-15.

MASS INVENTORIES

When isotopic atomic weights are available, the EDIT Module can, if requested by the user, calculate the mass inventories in the problem.* The default is to calculate the inventory of each isotope in each of the zones used by the SOLVER Module. Optionally, mass inventories for each of the EDIT zones may be calculated.

The input variable for the mass edit in Block-VI is “MASSED.” The allowed input values are shown in Table 4.1.

Table 4.1 MASSED Input Values

MASSED	Action
0	none
1	Edit by Solver zones ^a
2	Edit by Edit zones ^b
3	Edit by both Solver and Edit zones

- a. One is the default if there is any other Block-VI input. If the block is empty, then zero is the default.
- b. The default for the edit zones is the coarse mesh.

The isotopic atomic weights must be available to the Edit Module for the mass edit to work. They may be input in the Block-IV ATWT array, or they may be present in the cross-section library file. In either event, the atomic weights are written to the NDXSRF standard interface file, from whence the Edit Module acquires them.

*If the problem geometry has an infinite lateral dimension, the masses (and volumes) are given per unit height.

For problems in which the fine mesh mixing option is used, the MASSED input values are changed slightly as shown in Table 4.2.

Table 4.2 MASSED Input Values for Fine Mesh Mix Problem

MASSED	Action
0	none
1	Total inventory for the whole problem ^a
2	Inventories by Edit zones ^b
3	Inventories for both Total and Edit zones

- a. One is the default if there is any other Block-VI input. If the block is empty, then zero is the default.
- b. For linked problems, a Block-VI EDZONE array is required since there is no coarse mesh default for the edit zones. The coarse mesh and the fine mesh are identical in linked problems.

EDIT CHAPTER INDEX

A

Adjoint	
Edit of	4-15

E

Edits	
Energy specifications.....	4-10
Reaction rates	
From cross sections	4-11
From user defined response functions	4-13
Spatial specifications	4-9
Energy groups	
Broad groups in edits.....	4-10

F

Files	
Output, control of	4-15

G

Group collapse	
In edits	4-10

R

Response function	
Edits of.....	4-13

FREE FIELD INPUT REFERENCE

Transport Methods Group, CCS-4
Los Alamos National Laboratory

5

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	5-3
LIST OF TABLES.....	5-5
INTRODUCTION	5-7
FREE FIELD DEFINITIONS	5-9
Arrays	5-9
Numeric Data Items	5-9
Character Data Items	5-10
Blocks.....	5-10
Strings.....	5-10
Comments.....	5-10
Operators	5-10
FREE FIELD INPUT DETAILS	5-13
Free-Field Input.....	5-13
User-Specified Input Formats	5-16
U Operator:.....	5-16
V Operator:.....	5-17
FIXED-FIELD FIDO INPUT DETAILS	5-19
Fixed-Field FIDO Input	5-19
REFERENCES	5-23
FREE FIELD CHAPTER INDEX.....	5-24

LIST OF TABLES

Table 5.1: Free Field Data Operators.....	5-15
Table 5.2: Special Fixed Field Data Operators.....	5-21



INTRODUCTION

This chapter serves as the reference manual for the free field input format as implemented in PARTISN^{TM*}. First is a section defining terms and concepts. That is followed by a detailed reference section for the input rules, syntax, and operators for the free field input.

Lastly is a description of the alternate, fixed field, FIDO format. The alternate, fixed field FIDO format is an option for card input cross sections in PARTISN.

* - PARTISN is a trademark of the Regents of the University of California, Los Alamos National Laboratory

FREE FIELD DEFINITIONS

There are four basic input quantities in the free field input; they are ARRAY, DATA ITEM, BLOCK, and STRING. Each of these is described below along with the concept of an input operator.

Arrays

The “Array” is the most basic concept in the input. Data are given to the code by placing data items in an “Array.”

The favored input form (there are several) is one very similar to NAMELIST. Each input array has a unique name. To make an input to an array, one simply spells out the array name, appends an equal sign, and follows that with the data items to be entered into the array. For example, input for the x distribution of the volumetric source, for which the unique array name is SOURCX, could look like:

```
SOURCX= 0 0 0 1.1 1.1 0 0 0 0 0
```

The above input would enter source values of zero for the first three intervals, 1.1 for the next 2 intervals, and then fill the rest of the ten positions in the array with zero.

Unlike NAMELIST, however, an array name CANNOT use a subscript. The operator S, described later, may be used for this addressing function.

Data items within an array are separated by blanks or commas. In general, blanks may be used freely throughout except within a data item, within an array name, or between an array name and its equal sign.

Numeric Data Items

Numeric data items follow a FORTRAN input convention. For example, all of the following are valid entries for the number ten:

```
10, 1.0+1, 1E1, 10.0
```

If a decimal point is not entered, it is assumed to be after the right-most digit.

Some arrays expect integer values for input. For such arrays, any input values containing a decimal point will be truncated.

Character Data Items

Character data items follow a FORTRAN variable name convention in that they are composed of up to eight characters, the first of which must be alphabetic with the rest alphanumeric. However, special characters or blanks may be included if the data item is surrounded by double quotes. Operators may NOT be used with character data items.

Blocks

Arrays are entered in groups called blocks. A block consists of one or more arrays (in any order) followed by the single character T. Thus T is the block delimiter.

Strings

Arrays may need to be entered in smaller pieces called strings. Strings are delimited with a semicolon(;). When there is matrix or other 2-d input, strings are frequently used to input information by row rather than for the whole 2-d array at once. The code dictates this; the user has no choice. The user is made aware of which arrays require string input through use of a certain notation, described in the input array descriptions.

Comments

A slash (/) may be used to enter comments in the input stream. After a slash is read, no further processing of that card-image is done.

Operators

Several data operators are available to simplify the input. Most of these operators are FIDO¹ operators, but several are new and represent extensions to FIDO.

In free-field the data operators are specified in the general form

$$n \ O \ d$$

where:

n is the “data numerator,” an integer, or a blank;

O is any one of several “data operators” described in Table 5.1; and

d is a “data entry” (may be blank for some operators).

Note: The “data operator” character must be appended to the “data numerator.”

Using operators, the SOURCX input described above could more succinctly be given as:

SOURCX= 3r 0 2r 1.1 f0

Note that the operators for FIDO-like repeat and fill were used and were appended directly to the data numerator. In general, all the FIDO operators may be used in numeric entry.

A complete list of the valid operators is given in Table 5.1, "Free Field Data Operators," on page 5-15.

FREE FIELD INPUT DETAILS

This section describes the various rules, restrictions, and options available to the user when creating the input for PARTISN. First are described the details associated with free-field form of input. Next is presented the information needed for user-specified input forms. This is followed by information for fixed-field FIDO input.

Throughout this section we will use the term *card* or *card-image* to denote a single record or line of characters. Thus, an 80-column card (or card-image) refers to a line containing 80 columns into which ASCII characters may be written one per column.

Free-Field Input

1. Card-Image Ground Rules
 - a. Eighty (80) columns available.
 - b. No special columns; that is, no column is treated any differently than any other column.
2. Delimiters (Separators) and Terminators
 - a. Data Item Delimiter (Separator): one or more blanks, a comma, or end of card:

Note: Hereafter, when an item is referred to as being delimited, for example, delimited T, it means that the item must be separated from other data items by a blank, comma, or end of card.
 - b. Card Terminator: Slash (/), delimiting not required. All entries on a card beyond the slash are ignored.
 - c. Block Terminator: Delimited T. Information beyond the T on the card will be ignored.
 - d. Array Terminator: New array name or Block Terminator.
 - e. String Delimiter: Semicolon (;), delimiting not required on ;.
 - f. String Terminator: Semicolon or new array name or Block Terminator.
 - g. Data Item Terminator: Data item delimiter (Separator) or any of the above Terminators.
3. Numerical Data Item Ground Rules
 - a. Must not contain embedded blanks.

- b. Must not contain any nonnumeric characters except for E (for exponent), decimal point, or plus and minus signs.
4. Character Data Item Ground Rules
 - a. Must begin with alphabetic character [see c. below for exception] and may contain from one to eight characters.
 - b. Must not contain any of the following characters: =, \$, *, blank, comma, slash, semicolon, double quote ("). [See c. below for exception.]
 - c. Character data words may be delimited with double quotes to override a. and b. restrictions. For example, PU/239 is not allowed, but "PU/239" is allowed. Remember the eight character limit.
5. Array Identification and Ordering
 - a. To identify an array for which data entries are to be made, one simply enters the appropriate array name, followed by an equal (=) sign (no space between name and =) and then enters the desired data, for example, CHI= 0.95, 0.10 0.05 0.0 .
 - b. Within a given Block, arrays may be entered in any order.
6. Block Identification and Ordering
 - a. No explicit Block identification is required. Array identification is sufficient to tell the code which Block is involved. Recall, however, that a Block Terminator (delimited T) must be entered when all input arrays for a given Block have been entered.
 - b. Blocks must be ordered.
7. Input Data Operators.

Several data operators are available to simplify the input. Most of these operators are FIDO operators, but several are new and represent extensions to FIDO.

IMPORTANT NOTE: The data operators can *only* be used with arrays containing integer, real, or a combination of integer/real data entries. They are NOT usable with arrays that may contain character data items.

In free-field the data operators are specified in the general form

$$n \ O \ d$$

where:

n is the "data numerator," an integer, or blank;
O is any one of several "data operators" described in Table 5.1; and
d is a "data entry" (may be blank for some operators).

NOTE: When a “data numerator” is required with a “data operator,” there must be no space between the data numerator and the data operator. There may be any number of blanks between the data operator and the “data entry” if the latter is required.

In entering data using data operators, it is convenient to think of an index or pointer that is under the control of the user and which specifies the position in the data string into which the next data item is to go. The pointer is always positioned at string location number 1 when either an array identifier or a string terminator (;) is entered.

In the table below an entry of - for either the data numerator or data entry indicates that the item is not required for the particular data operator.

Table 5.1 Free Field Data Operators

Data Numerator	Data Operator	Data Entry	Remarks
n	R	d	REPEAT OPTION: Enter the data entry d n successive times in the current data string. Example: 3R 0.0 → 0.0 0.0 0.0
n	I	d	LINEAR INTERPOLATE OPTION: Enter the value d into the data string followed by n interpolated entries equally spaced between d and the next <u>data entry</u> . Allowed for both real and integer data although the spacing between interpolated integer data points must be integer. Example: 3I1, 5 → 1 2 3 4 5 but 3I1, 4 will cause an error if the array data type is integer.
n	L	d	LOGARITHMIC INTERPOLATE OPTION: The effect is the same as that of “I” except that the resulting interpolates are equally separated in log-space.
-	F	d	FILL OPTION: Fill the remainder of the data string with the value d.
n	Z	-	ZERO OPTION: Enter the value zero in the data string n successive times. Example: 4Z 0 → 0 0 0 0
n	S	-	SKIP OPTION: Causes the pointer to skip n positions in the current string leaving the data values in those positions unchanged.
n	Q	m	SEQUENCE REPEAT OPTION 1: Enter the last m data entries into the current string in sequence n successive times. Example: 1 2 3 1 2 3 1 2 3 can be input as 1 2 3 2Q3.

Table 5.1 Free Field Data Operators (Cont.)

Data Numerator	Data Operator	Data Entry	Remarks
n	G	m	SEQUENCE REPEAT OPTION 2: Same effect as “Q” except the sign of each entry in the sequence is reversed each time the sequence is repeated. Example 1 2 -1 -2 1 2 can be input as 1 2 2G2.
n	N	m	SEQUENCE REPEAT OPTION 3: Same effect as “Q” except the order of the sequence is reversed each time the sequence is entered. For example, 4 5 6 6 5 4 4 5 6 can be input as 4 5 6 2N3.
n	M	m	SEQUENCE REPEAT OPTION 4: Same effect as “N” except the sign of each entry in the sequence is reversed each time the sequence is entered. Example: 1 2 3 -3 -2 -1 1 2 3 can be input as 1 2 3 2M3.
n	Y	m	STRING REPEAT OPTION: Enter the preceding m strings of data into the current array n successive times. (For multistring arrays only.)
n	X	-	COUNT CHECK OPTION: Causes code to check the number of data items entered into the current string to see if the number of items equals n. If count is not correct, an error message will be printed, an attempt will be made to continue processing all remaining input, and then the problem will be halted. (Error diagnostic aid.)
n	C	d	SEQUENCE MULTIPLY OPTION: Enter the last n data entries, multiplied by d, into the current data string.

User-Specified Input Formats

If desired, input data for an array can be provided in a format specified by the user. To specify the format for the input data to an array, the user can use the characters U or V as follows:

U Operator:

1. Enter the array identification and follow this with a delimited U.

2. On the next card-image enter the desired format enclosed in parentheses anywhere in columns 1-72.
3. On the next and succeeding cards enter the data using ordinary FORTRAN rules.

Example:

```
CHI= U  
(6E12.5)  
data in 6E12.5 format
```

V Operator:

Has same effect as U except the desired format is not entered; instead the format read in the last preceding U array is used.

FIXED-FIELD FIDO INPUT DETAILS

Cross sections may be input in a fixed-field FIDO format. But none of Blocks-I through VI input contain any fixed-field FIDO format. Details of the format are given below.

Fixed-Field FIDO Input

1. Card-Image Ground Rules
 - a. Seventy-two (72) columns available.
 - b. Each card divided into six “fields” of 12 columns each.
 - c. Each 12-column “field” subdivided into three subfields containing 2, 1, and 9 columns, respectively. Hereafter these subfields will be referred to as Subfield 1 (the first two columns of each field), Subfield 2 (the third column in each field), and the Data Subfield (the remaining nine columns in each field).
2. Delimiters (Separators) and Terminators
 - a. Data Item Delimiter (Separator): Field and subfield column boundaries.
 - b. All entries following the slash on the card are ignored.
 - c. Block Terminator: T in second subfield of any field. All entries beyond the T on that card are ignored.
 - d. Array Terminator: New array identified in first and second subfields of next field, or a Block Terminator.
 - e. String Delimiter: Semicolon (;) in second subfield of any field. Data subfield of that field is ignored.
 - f. String Terminator: Semicolon or Array Terminator or Block Terminator.
3. Numerical Data Ground Rules
 - a. Standard FORTRAN convention.
 - b. Data items entered in third subfield (the data subfield) in each field only.
4. Hollerith Data Item Ground Rules

Character data not allowed.
5. Array Identification and Ordering

- a. To identify an array for which data are to be entered in the fixed-field FIDO format, one simply enters the array *number* (integer, ≤ 99) in the first subfield of any field followed by the array-type indicator (array purpose character) in the second subfield. If the array data is integer (fixed point), the array-type indicator is the dollar sign (\$); if the array data is real (floating point), the array-type indicator is an asterisk (*). The third subfield is left blank.
 - b. Arrays may be entered in any order within a given Block.
6. Block Identification and Ordering
 - a. No explicit Block identification required. Array identification is sufficient to tell the code which Block is involved. Recall that a Block Terminator (T) must be entered when all input arrays for a given block have been entered.
 - b. Blocks must be ordered.
 7. Input Data Operators

The fixed-field FIDO data operators consist of a subset of the operators used in the free-field input shown in Table 5.1 plus the special fixed field operators shown in Table 5.2. Only T, *, R, -, +, and Z may be used. In fixed-field FIDO usage of these operators, however, the following rules must be observed:

- a. the “data numerator,” if required, must be entered in the first subfield of a field;
- b. the “data operator” must be entered in the second subfield of a field; and
- c. the “data entry.” if required, must be entered in the third subfield of a field.

Table 5.2 Special Fixed Field Data Operators

Data Numerator	Data Operator	Data Entry	Remarks
n	*		FLOATING ARRAY IDENTIFIER. The unique array number is n. Precedes the data in the array.
n	+	d	POSITIVE EXPONENTIATION. This will enter a single number with magnitude $d \times 10^{+n}$, i.e., d is the mantissa and n is the tens exponent. If no decimal point is present in d, it will be assumed to be at the far right.
n	-	d	NEGATIVE EXPONENTIATION. This will enter a single number with magnitude $d \times 10^{-n}$, i.e., d is the mantissa and n is the tens exponent. If no decimal point is present in d, it will be assumed to be at the far right.

REFERENCES

1. W. A. Rhoades and R. L. Childs, "An Updated Version of the DOT4 One- and Two-Dimensional Neutron/Photon Transport Code," Oak Ridge National Laboratory report ORNL-5851, (July 1982).

FREE FIELD CHAPTER INDEX

A

Array	
Definition of	5-9

B

Block	
Definition of	5-10

C

Comments	
Embedded in input lines	5-10

D

Data item	
Character, definition of.....	5-10
Numeric, definition of	5-9
Data operators	
Purpose of.....	5-10
Table of.....	5-15
Usage form	5-10
Delimiters	
In free field input	5-13

F

FIDO	5-19
Fixed field format	5-19
Free field input	
Delimiters	5-13
Syntax details.....	5-13
Terminators.....	5-13
User specified format	5-16

O

Operators

In input, see also "Data operators"	5-10
Table of.....	5-15

S

String

Definition of	5-10
---------------------	------

T

Terminators

In free field input	5-13
---------------------------	------

CROSS-SECTION LIBRARIES

Transport Methods Group, CCS-4
Los Alamos National Laboratory

6

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	6-3
LIST OF TABLES.....	6-5
INTRODUCTION	6-7
INPUT OF THE BASIC CROSS-SECTION LIBRARY.....	6-9
ISOTXS and GRUPXS Standard Interface Files	6-9
Card-Image Libraries in Los Alamos, ANISN, or Free Field Format	6-9
Ordering of Cross Sections Within a Cross-Section Table.....	6-10
Card-Image Data Formats	6-11
Cross-Section Table Title Cards	6-11
Anisotropic Scattering and the Ordering of Cross-Section Tables.....	6-11
Binary Form of Card-Image Libraries (the BXSLIB file)	6-12
XSLIBB Card-Image Library File	6-12
MACRXS and SNXEDT Cross-Section Files	6-13
The MACBCD Card-Image Cross-Section Library	6-13
The Los Alamos MENDF Cross-Section Libraries	6-13
The Los Alamos MENDF5G Gamma Cross-Section Library	6-14
The XSLIBE and XSLIBF Material Cross-Section Libraries.....	6-14
COUPLED NEUTRON-GAMMA CROSS SECTIONS	6-17
CREATING FILES WITH DIFFERENT FORMATS.....	6-21
BALANCING THE CROSS SECTIONS	6-23
REFERENCES	6-25
CROSS SECTION CHAPTER INDEX	6-26

LIST OF TABLES

Table 6.1:	Cross-Section Ordering In A Card-Image Library	6-10
Table 6.2:	Arrangement of Data in a Coupled Neutron-Gamma Library Table	6-18
Table 6.3:	Coupled Cross-Section Table Example	6-19



INTRODUCTION

In this chapter, we address the details of the many ways that multigroup cross sections can be supplied to PARTISN^{TM*}. We do NOT address the many issues involved in the processing of cross-section data into the multigroup format. Although the end user needs to be somewhat knowledgeable in that area, we feel that is outside the purview of this document.

The code cannot only read various cross-section library formats, but also has the capability to write several output formats. A few pages are devoted to these output formats and why one might want to use them.

Most of this chapter is concerned with the format and ordering of the data within each of several possible library files. This includes one section on how to order cross sections in a coupled library, such as a neutron-gamma library.

Then lastly, there is a small section about enforcing balance in the cross-section sets. This section touches briefly, but necessarily, on the production of the multigroup cross sections.

* - PARTISN is a trademark of the Regents of the University of California, Los Alamos National Laboratory

INPUT OF THE BASIC CROSS-SECTION LIBRARY

The general procedure for generating the macroscopic cross sections appropriate to each zone in the problem is to begin with a basic library containing multigroup cross-section data for isotopes. This section describes the allowable forms that these libraries can take.

ISOTXS and GRUPXS Standard Interface Files

Either of the standard interface files ISOTXS or GRUPXS can be used for providing the basic, multigroup cross sections for isotopes. ISOTXS is an isotope-ordered, binary file while GRUPXS is a group-ordered binary file. These are standard interface files as described by the Committee on Computer Code Coordination.^{1,2} By default, the file wide vector chi (fission fractions) from these files will be used unless overridden by the zone dependent CHI in Block-V.

If the basic library of isotope cross sections is an ISOTXS file, the user enters LIB=ISOTXS in the Block-III input; if the library is a GRUPXS file, the user enters LIB=GRUPXS. If LIB=ISOTXS, the cross sections, by default, must reside on a file named ISOTXS which must exist at the time of code execution. The user can override the name ISOTXS by setting LIBNAME=*filename* in the input, where *filename* is the name of the ISOTXS file. For LIB=GRUPXS, the same conditions apply.

Card-Image Libraries in Los Alamos, ANISN, or Free Field Format

The basic multigroup cross sections for isotopes can be provided in a card-image library whose form is referred to as Los Alamos, ANISN, or Free Field. This library form consists of a collection of cross-section tables. Each of these cross-section tables contains the full set of multigroup cross sections for one Legendre scattering order for one isotope. The ordering of cross sections within a cross-section table, the ordering of cross-section tables to form the library, and other details and user options are described below.

The user specifies that the library of cross sections is to be such a card-image library by entering either LIB=ODNINP or LIB=XSLIB. If LIB=ODNINP, the library card-images are physically located within the input for the code between the input for Block-III and the input for Block-IV. If LIB=XSLIB, the library card-images are physically located on a file named XSLIB, which must exist at the time of code execution.

The user may also specify a card-image library by entering LIB=*filename* where *filename* is any name that the user chooses other than any of the following: ISOTXS,

GRUPXS, MACRXS, MACBCD, BXSLIB, XSLIBB, MENDF, or MENDFG. The library card-images are then physically located on a file named *filename*.

Ordering of Cross Sections Within a Cross-Section Table

The Los Alamos, ANISN or FIDO card-image library form all assume that each cross-section table in the library contains an array of cross sections of NGROUP rows each containing IHM positions. The cross-section type for each group is determined by its position as shown in Table 6.1. Positions are specified relative to the positions of the total cross section σ_t (position IHT) and the within-group scattering cross section $\sigma_{g \rightarrow g}$ (position IHS). Note that the values of IHM, IHT, and IHS are user input values in Block-III.

Each cross-section table contains the cross sections for one Legendre scattering order for one isotope as IHM*NGROUP data entries. A cross-section table begins on a new card-image and the data are entered continuously beginning with IHM entries for group 1, followed by IHM entries for group 2, etc.

Table 6.1 Cross-Section Ordering In A Card-Image Library

Position in the Row	Cross-Section Type For Group g	Collective Designation
.	.	Edit Positions, Optional
.	.	
.	.	
IHT-2	σ_a	Principal Cross Sections
IHT-1	$v\sigma_f$	
IHT ^a	σ_t	
.	.	Upscatter Cross Sections
.	.	
.	.	
IHS-2	$\sigma_{s(g+2 \rightarrow g)}$	
IHS-1	$\sigma_{s(g+1 \rightarrow g)}$	
IHS ^a	$\sigma_{s(g \rightarrow g)}$	Self

Table 6.1 Cross-Section Ordering In A Card-Image Library

Position in the Row	Cross-Section Type For Group g	Collective Designation
IHS+1	$\sigma_s(g-1 \rightarrow g)$	Downscatter Cross Sections
IHS+2	$\sigma_s(g-2 \rightarrow g)$	
.	.	
.	.	
.	.	
IHM ^a	.	Last Position

a. IHM and IHS are user input in Block-III

Card-Image Data Formats

The cross-section data may be entered on the card-images in one of four data formats, the traditional Los Alamos format, another similar format with more accuracy, the fixed-field FIDO format, or the free-field format. The user selects the desired format through the IFIDO input parameter in the Block-III input.

In the traditional Los Alamos format (IFIDO= 0), also called the DTF format, the data are entered on the card-images in 6E12 format.

For more accuracy a similar format (IFIDO= -1) is provided, for which the data are entered on the card-images in 4E18 format.

In the fixed-field FIDO format (IFIDO= 1), sometimes called the ANISN format, the data are entered on the card-images using the fixed-field FIDO format described in "FIXED-FIELD FIDO INPUT DETAILS" on page 5-19. When this format is used, each cross-section table must be terminated with the "T" terminator as described there.

In the free-field format (IFIDO= 2), the data are entered on the card-images in free-field format as described in the "FREE FIELD INPUT DETAILS" on page 5-13. When this format is used, each cross-section table must be terminated with the "T" terminator described there.

NOTE: For fixed-field FIDO or free field cross sections, neither an array name (or number) nor an array identifier is needed with the cross-section data.

Cross-Section Table Title Cards

A single title card may optionally be attached to the front of each cross-section table, if desired. This option is controlled by the input parameter, ITITL in the Block-III input.

Anisotropic Scattering and the Ordering of Cross-Section Tables

In the code, it is assumed that the scattering transfer probability can be represented by the finite Legendre polynomial expansion of equation (14) on page 8-23, which, in multigroup notation, becomes

$$\sigma_s(g' \rightarrow g, \mu_o) = \sum_{n=0}^{ISCT} \frac{2n+1}{4\pi} P_n(\mu_o) \sigma_s^n(g' \rightarrow g) ,$$

where $\mu_o \equiv \underline{\Omega}' \cdot \underline{\Omega}$, the scattering angle and ISCT is the desired Legendre order of anisotropy in the transport calculation (input in Block-V). If ISCT>0, additional tables of cross sections must be supplied in order to provide the higher order scattering cross sections $\sigma_s^n(g' \rightarrow g)$ needed for the Legendre expansion.

When using the card image library, the first cross-section table for an isotope contains the P_0 , or isotropic, multigroup cross sections ordered as shown in Table 6.1. The next cross-section table provides the P_1 multigroup cross sections with the same ordering; the next table contains the P_2 multigroup cross sections, etc., all for the same isotope. It should be noted that, for high Legendre order cross-section tables, only the scattering cross sections are used. The first IHT rows for each group are ignored in the $P_L(L>0)$ tables and the data values in these positions are usually input as 0.0. The number of tables per isotope can vary with each isotope. The number of cross-section tables per isotope is provided in the input array NTPI in Block-III. If the NTPI array is not provided, the code will assume that the card-image library contains MAXORD + 1 cross-section tables for each isotope, where MAXORD is an input parameter in Block-III.

Note that the library may contain scattering data for up to a MAXORD order of anisotropy, but the actual transport calculation can be performed assuming an ISCT order of anisotropy so long as $ISCT \leq MAXORD$.

Binary Form of Card-Image Libraries (the BXSLIB file)

The processing of large, card-image, ASCII libraries can be relatively time-consuming, especially if the library is in FIDO format. The binary form of the card-image library can be processed much more rapidly. By entering LIB= BXSLIB, the user can instruct the code to use the binary form of the card-image, isotopic library (the binary file named BXSLIB) as the input for the basic cross-section data.

Use of LIB= BXSLIB requires that the appropriate binary form of the library exists and is available to the code at the time of execution. To create the BXSLIB file, the user makes his initial execution with the card-image library (LIB= XSLIB or LIB= ODN-INP) as previously described. Then, by setting the input parameter SAVBXS= 1 in the Block-III input, the user can instruct the code to create the binary BXSLIB file and to retain this file after execution of the Input Module. For Los Alamos users, if LIB= MENDF, (see “The Los Alamos MENDF Cross-Section Libraries” on page 6-14) a BXSLIB file is always created and retained. The user can then save this BXSLIB binary file and use it for subsequent runs in place of the BCD library. It should be noted that in addition to the actual cross-section data, the BXSLIB file will contain any and all other information specified in the table labelled “Text Cross-Section Library Format” in the Block-III input description of any of the User’s Guides. The information placed there will be that provided in the originating LIB= ODNINP or LIB= XSLIB run. Also included on the file is the input atomic weights, if they were input via card input in Block-IV. The file description for the BXSLIB binary file is provided in the chapter “FILE DESCRIPTIONS”. See “BXSLIB” on page 11-37.

XSLIBB Card-Image Library File

By entering LIB=XSLIBB in the Block-III input, the user instructs the code to use the specialized isotopic cross-section file named XSLIBB. XSLIBB is an ASCII, card-image version of the BXSLIB file described previously. The principal advantage of an XSLIBB file is that it is an ASCII file and is thus both eye-readable and exportable. Since it can also contain other information such as NAMES, EDNAME, NTPI, CHIVEC, VEL, ATWT, and EBOUND, it provides a very useful form of a cross-section library. Use of LIB=XSLIBB requires that the appropriate form of the card-image file exists with the name XSLIBB and is available to the code at the time of execution.

The creation of an XSLIBB file is controlled by the WRITMXS parameter in Block-III as described on page 6-21.

MACRXS and SNXEDT Cross-Section Files

By entering LIB= MACRXS in the Block-III input, the user can instruct the code to use the code-dependent interface files MACRXS and SNXEDT together with the standard interface files NDXSRF and ZNATDN¹² without referring to a basic library of multi-group isotope cross sections. These four files contain cross sections and other information pertaining to the materials created from the original isotopes. (A more detailed discussion of the MACRXS and SNXEDT file preparation process is provided in “INTERFACE FILES USED IN MIXING” on page 7-17 and file descriptions for MACRXS and SNXEDT are given in the chapter “FILE DESCRIPTIONS” starting on page 11-1.) This procedure circumvents the sometimes time-consuming process of re-

creating these files when a series of code calculations are being made on the same basic problem.

If the user enters LIB= MACRXS, it is understood that the MACRXS, SNXEDT, NDXSRF, and ZNATDN files must have been previously created and saved and, further, that these files must be available to the code at the time of execution as follows:

- a. MACRXS is required if the SOLVER module is to be executed, and
- b. SNXEDT, NDXSRF, and ZNATDN are required if the EDIT module is to be executed.

The MACBCD Card-Image Cross-Section Library

By entering LIB=MACBCD in the Block-III input, the user instructs the code to use the specialized material cross-section file named MACBCD. MACBCD is an ASCII, card-image version of the MACRXS described in the preceding section. Use of LIB=MACBCD requires that the appropriate card-image file exists with the name MACBCD and that it is available to the code at the time of execution.

The creation of a MACBCD file is controlled by the WRITMXS parameter in Block-III as described later in the chapter.

The Los Alamos MENDF Cross-Section Libraries

At Los Alamos National Laboratory multigroup, isotopic cross-section libraries are maintained as random access public files available to users on some of the Laboratory's mainframe computers.

The public file named MENDF5 is derived from ENDF-B/V nuclear data evaluations. To use this binary library, or a library constructed in the same format as MENDF5, the user enters LIB=MENDF in the Block-III input. The code will seek a file named MENDF in the user's local file space and, if such a file exists, will use it. If a file named MENDF does not exist in the local file space, the code will extract the MENDF5 public file and will use it instead.

The public file named MENDF6 is derived from ENDF-B/VI nuclear data evaluations. To use this binary library, or a library constructed in the same format as MENDF5, the user enters LIB=MENDF and LIBNAME=MENDF6 in the Block-III input. The code will seek a file named MENDF6 in the user's local file space and, if such a file exists, will use it. If a file named MENDF6 does not exist in the local file space, the code will extract the MENDF6 public file and will use it instead.

The number of isotopes on the MENDF library must be entered using the NISO array of Block-I.

The appropriate fission fractions may be specified using the NTICHI input parameter in Block-III, or by using the CHIVEC input array of Block-III, or by using zone-dependent chi's in the Block-V input.

When using a MENDF file, isotopes are identified by a floating point number called a ZAID. The ZAID is of the form ZZAAA.NN where ZZ is the atomic number, AAA is the (three-digit) mass number, and NN is a two-digit number specifying a particular version of the cross sections for each given isotope. A listing of the current ZAIDs available on MENDF5 and MENDF6 can be obtained from the Radiation Transport Group at Los Alamos National Laboratory.

The Los Alamos MENDF5G Gamma Cross-Section Library

Similar to the MENDF5 library at Los Alamos described in the preceding section, a multigroup, isotopic companion gamma-ray (photon) library is maintained. MENDF5G contains only neutron-induced photon-production and photon-interaction data. To access this library for gamma (photon)-only calculations, the user specifies LIB=MENDFG in Block-III of the input and the LNG (Last Neutron Group) parameter of Block-III must be set to zero.

The number of isotopes on the MENDFG library must be entered using the NISO array of Block-I.

Isotopes are referenced by their ZAIDs as described in the preceding section.

The XSLIBE and XSLIBF Material Cross-Section Libraries

The code can use card-image material cross-section libraries named XSLIBE and XSLIBF by setting LIB=XSLIBE or XSLIBF in Block-III of the input. These two files are ASCII, card-image files of the XSLIB type containing the material macroscopic cross sections taken from the MACRXS binary file. The format of these card-image files is the Los Alamos or ANISN format described on page 6-9 of this chapter. XSLIBE is formatted in the Los Alamos 6E12 card-image format while XSLIBF is formatted in the fixed-field FIDO format.

These files can be created by the code using the WRITMXS parameter in Block-III as described on page 6-21.



COUPLED NEUTRON-GAMMA CROSS SECTIONS

These codes can solve coupled neutron-gamma problems in which neutron interactions with matter produce a source of gamma rays (photons). The simultaneous solution of the neutron-gamma transport problem can be effected by simply providing a coupled neutron-gamma cross-section library or set. In such a coupled set the gamma energy groups are treated as if they were the lowest energy neutron groups.

As an example, a 42-group coupled set ($NGROUP = 42$) might have 30 neutron groups ($LNG = 30$) followed by 12 gamma groups. If there is no upscatter in the system, a coupled set can be provided in the form of a card-image library with $IHS = IHT+1$ and $IHM = IHT+NGROUP$. In this form neutrons appear to “downscatter” into the gamma-ray groups as a result of gamma production resulting from neutron interactions but gamma-rays do not “upscatter” into neutron groups, i.e., there is no neutron production via photon-neutron, or gamma-n, reactions. Using the card-image form of a coupled library with cross sections ordered as shown in Table 6.2, “Arrangement of Data in a Coupled Neutron-Gamma Library Table,” on page 6-18 (for no upscatter), the isotopic cross sections for each Legendre order of scatter carry data arranged as shown in Table 6.3. That table shows the contents of a card-image cross-section table for a 7-group coupled set (4 neutron, 3 gamma groups).

Table 6.2 Arrangement of Data in a Coupled Neutron-Gamma Library Table

		GROUP				
ROW	1 2 . .	LNG	LNG+1	.	NGROUP	
1		PRINCIPAL CROSS SECTIONS AND EDIT POSITIONS				
2						
⋮						
IHT		NEUTRON PRODUCTION VIA NEUTRON SCATTER		GAMMA PRODUCTION VIA GAMMA SCATTER		
IHS		NOT USED		GAMMA PRODUCTION VIA NEUTRON SCATTER		
⋮						
IHS+LNG-1						
IHS+LNG						
⋮						
IHM=						
IHT+NGROUP						

Table 6.3 Coupled^a Cross-Section Table^b Example

	NEUTRON GROUPS				GAMMA GROUPS		
	1	2	3	4	5	6	7
1	$\sigma_a(1)$	$\sigma_a(2)$	$\sigma_a(3)$	$\sigma_a(4)$	$\sigma_a(5)$	$\sigma_a(6)$	$\sigma_a(7)$
2	$\nu\sigma_f(1)$	$\nu\sigma_f(2)$	$\nu\sigma_f(3)$	$\nu\sigma_f(4)$	0	0	0
3	$\sigma_t(1)$	$\sigma_t(2)$	$\sigma_t(3)$	$\sigma_t(4)$	$\sigma_t(5)$	$\sigma_t(6)$	$\sigma_t(7)$
4	$\sigma(1 \rightarrow 1)$	$\sigma(2 \rightarrow 2)$	$\sigma(3 \rightarrow 3)$	$\sigma(4 \rightarrow 4)$	$\sigma(5 \rightarrow 5)$	$\sigma(6 \rightarrow 6)$	$\sigma(7 \rightarrow 7)$
5	0	$\sigma(1 \rightarrow 2)$	$\sigma(2 \rightarrow 3)$	$\sigma(3 \rightarrow 4)$	$\sigma(4 \rightarrow 5)$	$\sigma(5 \rightarrow 6)$	$\sigma(6 \rightarrow 7)$
6	0	0	$\sigma(1 \rightarrow 3)$	$\sigma(2 \rightarrow 4)$	$\sigma(3 \rightarrow 5)$	$\sigma(4 \rightarrow 6)$	$\sigma(5 \rightarrow 7)$
7	0	0	0	$\sigma(1 \rightarrow 4)$	$\sigma(2 \rightarrow 5)$	$\sigma(3 \rightarrow 6)$	$\sigma(4 \rightarrow 7)$
8	0	0	0	0	$\sigma(1 \rightarrow 5)$	$\sigma(2 \rightarrow 6)$	$\sigma(3 \rightarrow 7)$
9	0	0	0	0	0	$\sigma(1 \rightarrow 6)$	$\sigma(2 \rightarrow 7)$
10	0	0	0	0	0	0	$\sigma(1 \rightarrow 7)$

a. 4 Neutron groups, 3 Gamma groups

b. Detailed specifications:

NGROUP = 7

LNG = 4

IHT = 3

IHS = 4

IHM = 10



CREATING FILES WITH DIFFERENT FORMATS

It is frequently desirable to produce card-image, ASCII forms of cross-section libraries that are eye-readable and exportable. The Input Module provides the user with the capability to produce several card-image library forms through the use of the WRITMXS parameter in Block-III of the input.

By setting WRITMXS=XSLIBB, the code is instructed to create the ASCII file XSLIBB described on page 6-13. It can be used if the original cross-section library (as specified in the LIB= instruction) is ODNINP, XSLIB, MENDF, MENDFG, or BXSLIB.

By entering WRITMXS=MACBCD, the code will create the ASCII file MACBCD described on page 6-14. The MACBCD file can be created no matter what the form of the original cross-section library.

By entering WRITMXS=XSLIBE, the code will create the ASCII file XSLIBE containing the material cross sections in Los Alamos 6E12 format as described on page 6-9 and page 6-15. Similarly, WRITMXS=XSLIBF instructs the code to create the ASCII file XSLIBF containing the material cross sections in fixed-field FIDO format as described on page 6-9 and page 6-15. Either file can be created irrespective of the form of the original cross-section library (as specified in the LIB= instruction).

BALANCING THE CROSS SECTIONS

The cross-section balancing option, triggered by the input variable, BALXS, should rarely be used. Proper use of BALXS requires a thorough understanding of the cross-section processing that went into the preparation of the multigroup cross-section library and why they are not balanced.

WARNING! Improper use of BALXS can INVALIDATE your flux solution.

In order to understand what BALXS does, we first discuss what PARTISN requires of the cross-section set. When making the flux calculation for group g , it is important that the cross section for removal from the group be accurate. Removal from a group occurs either through absorption or through scattering to another group. The code implicitly gets the removal cross section, $\sigma_{r,g}$, from the total and self scattering cross sections:

$$\sigma_{r,g} = \sigma_{\text{total},g} - \sigma_{g \rightarrow g} \quad (20)$$

Notice that nowhere is the absorption cross section, $\sigma_{\text{abs},g}$, explicitly mentioned. It is NOT used directly or indirectly anywhere in the flux calculation. The flux calculation will give the correct results, given only correct values for $\sigma_{\text{total},g}$ and $\sigma_{g \rightarrow g}$ (and, of course, the group to group transfer cross sections).

However, in order to provide the out-scattering rate for the balance table, the absorption cross section IS used. Recall that in the cross sections for a group, we have only the inscattering cross sections from other groups and not the outscattering cross section. One could access all the cross-section groups in order to sum up the outscatter from the source group, but conventionally, a short cut is used. The outscatter term is formed using an effective outscatter cross section, $\sigma_{\text{out},g}$, calculated as shown in Eq. (21):

$$\sigma_{\text{out},g} = \sigma_{r,g} - \sigma_{\text{abs},g} \quad (21)$$

Now, consider the cross-section treatment for the (n,2n) reaction rate. Here we have, for each neutron lost to the source group, two neutrons appearing in other groups. Typically, this is handled by including the (n,2n) cross section in the total and the absorption positions of the source group, but then including a total of twice the (n,2n) cross section in the inscattering cross sections of the recipient groups. This procedure gives the correct removal cross section as represented by Eq. (20) and also accurately represents the inscattering source to the recipient groups. It also gives correctly the outscattering rate that shows in the balance table. However, the inscattering rate shown in the system balance table is formed from the inscatter cross sections and will include the (n,2n) produc-

tion. The total inscatter rate (i.e. summed over the energy groups) will then be larger than the total outscatter rate by the amount of the production rate from the (n,2n) reaction.

Normally, the cross sections should balance, that is, the absorption reaction rate plus the total scattering reaction rate should equal the total reaction rate:

$$\sigma_{abs,g} + \sum_{g'} \sigma_{g \rightarrow g'} = \sigma_{total,g} \quad (22)$$

Again, recall that the transfer cross sections, $\sigma_{g \rightarrow g'}$, that we have in the library are normally the inscattering cross sections to group g' and not the outscattering cross sections from group g as required by Eq. (22). So, the cross sections will not be balanced according to Eq. (22) if there is any (n,2n). Typically, the remedy is to redefine the absorption cross section (recalling that this will not affect the flux calculation) to be:

$$\sigma'_{abs,g} = \sigma_{total,g} - \sum_{g'} \sigma_{g \rightarrow g'} \quad (23)$$

Where we are now using the only transfer cross sections we have, i.e., the inscattering cross sections, for $\sigma_{g \rightarrow g'}$. Note that $\sigma'_{abs,g}$ is then no longer the physical absorption cross section when we have any (n,2n). Also note that this new “ $\sigma_{abs,g}$ ” will cause the total outscattering in the system balance table to be equal to the total inscattering even in the presence of (n,2n), and hence, you will get the true system balance there.

The bottom line is that if your cross-section processor supplies a balanced set with the proper total cross section, the proper scattering cross sections in the transfer matrix, and an absorption cross section satisfying Eq. (23), you should get the correct flux calculation and the correct balance calculated in the balance table even in the presence of (n,2n).

However, let us suppose that the cross-section purveyor supplies the true absorption cross section rather than that shown in Eq. (23) and that there is significant (n,2n) in the material. The cross-section set will not then be balanced. According to the earlier discussion, you will still get the correct flux solution, but the calculated system balance will be artificially off due solely to this peculiarity of the cross sections. The balance then is not a good measure of how well the iteration procedure went in determining the flux. You can use the input option, BALXS = -1, to replace the supplied $\sigma_{abs,g}$ with that calculated from Eq. (23), again recalling that changes in the absorption cross section will not affect the flux calculation, but will remove the artificial part of the imbalance.

If you use the other option, BALXS = +1, the self scattering cross section in Eq. (23) will be adjusted to balance the cross sections. But as this WILL change the flux solution

because you are then changing the removal cross section, you must know that you are correcting an error in the cross-section processing when you use this option. If you force cross-section balancing this way when it is inappropriate, you will get an INCORRECT flux solution.

REFERENCES

1. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
2. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physics Codes, Version III," Los Alamos Scientific Laboratory report LA-5486-MS (February 1974).

CROSS SECTION CHAPTER INDEX

C

Cross section library

ASCII, writing from code	6-21, 6-23
Binary, writing from code	6-12
BXSLIB	6-12
Card image libraries	6-9
Coupled neutron-gamma libraries	6-17
Details for inputting	6-1
GRUPXS	6-9
ISOTXS	6-9
MACBCD	6-13
MACRXS	6-13
MENDF5	6-13
MENDF5G	6-14
SNXEDT	6-13
XSLIB	6-9
XSLIBB	6-12
XSLIBE, XSLIBF	6-14

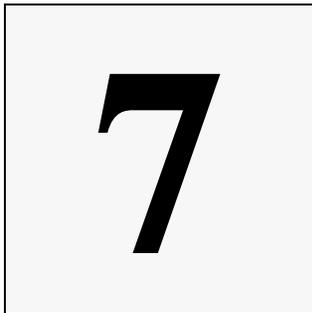
X

XSLIB

Format of	6-9
-----------------	-----

MATERIAL MIXING TUTORIAL

Transport Methods Group, CCS-4
Los Alamos National Laboratory



CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	7-3
LIST OF TABLES.....	7-5
REVIEW OF TERMINOLOGY	7-7
MIXING MATERIALS FROM “ISOTOPES”	7-9
ASSIGNING MATERIALS TO ZONES.....	7-11
ALTERNATIVE FORMS OF MIXING	7-13
Using Atomic Fractions or Weight Fractions (MATSPEC).....	7-13
Providing Atomic Weights to the Code (the ATWT Array)	7-15
INTERFACE FILES USED IN MIXING	7-17
REFERENCES	7-19
MIXING CHAPTER INDEX	7-14

LIST OF TABLES

Table 7.1: Example Specification of Materials	7-10
Table 7.2: Composition of Zones	7-11



REVIEW OF TERMINOLOGY

An understanding of the general procedure for mixing nuclides in these codes requires an understanding of the terminology used with the code. A review of five basic terms is provided below.

1. Fine Mesh - the Fine Mesh is the spatial solution mesh for the problem.
2. Coarse Mesh - the Coarse Mesh is a spatial superset of the Fine Mesh. Each Coarse Mesh contains one or more Fine Mesh intervals. Fine mesh widths are all the same within a given coarse mesh interval. No material discontinuities may occur within a coarse mesh interval.
3. Zone - the Zone is a spatial superset of the Coarse Mesh intervals. Each Zone contains one or more Coarse Mesh intervals. A Zone is characterized by a single set of macroscopic multigroup cross sections so that all fine mesh intervals within a zone have the same cross sections.
4. Material - a Material is composed of isotopes (or nuclides or mixes) whose “microscopic” cross sections exist on the cross-section library file to be used. The specification of materials requires knowing which isotopes and how much of each isotope goes into the Material. This information is provided the code through the MATLS= array in Block-IV of the input.
5. Material Assignments to Zones - the material composition of a Zone is made by assigning one or more Materials in specified amounts to each Zone through the ASSIGN= input array in Block-IV of the input. This assignment thus links a spatial portion of the physical problem model (the zone) to the macroscopic cross sections that are to be used in that spatially-defined zone.

MIXING MATERIALS FROM “ISOTOPES”

If “isotopic” cross sections are provided on a cross-section library, it is necessary to mix these isotopes, or nuclides, to create materials. The mixing instructions are provided either by (i) card-image input in Block-IV by means of the MATLS array and, optionally, the PREMIX array, the specifications of which are described in the User’s Guide, or by (ii) the standard interface files NDXSRF and ZNATDN.¹ If the NDXSRF and ZNATDN files are used, the term “zone” in the file descriptions of Ref. 1 and Ref. 2 must be replaced with the word “material” to be consistent with our terminology.

In this section we will provide a basic description of how to mix isotopes (or nuclides) to form materials using the MATLS array specification in Block-IV of the input. For now we will assume that it is desired to create materials by specifying the isotopes and their atom densities in each material. Later in this chapter will be described some alternatives to using atom densities.

Let us consider an example of a common way of specifying materials. Suppose we have a multigroup library of cross sections (in barns) for several nuclides including those whose identifiers are U235, U238, PU239, PU240, IRON, CHROME, NICKEL, OXYGEN, and SODIUM. It is desired to mix these nuclides into the following materials:

1. Stainless Steel (SS) at 8 gm/cm^3
[74% Iron, 18% Chromium, and 8% Nickel (percents are weight percents)],
2. Uranium Dioxide (UO₂) at 10 gm/cm^3
[0.3% U235 and 99.7% U238 (percents are atomic percents)],
3. Plutonium Dioxide (PUO₂) at 10 gm/cm^3
[80% PU239 and 20% PU240 (percents are atomic percents)],
4. Sodium (NA) at 0.8 gm/cm^3 .

Translating the above information into atom densities (atoms/barn-cm) gives the values shown in Table 7.1.

Table 7.1 Example Specification of Materials

Material Identifier	Isotope Identifier	Atom Density
SS	CHROME	1.6676E-2
	NICKEL	6.566E-3
	IRON	6.3832E-2
UO2	U235	6.7E-5
	U238	2.223E-2
	OXYGEN	4.460E-2
PUO2	PU239	1.7761E-2
	PU240	4.440E-3
	OXYGEN	4.4402E-2
NA	SODIUM	2.096E-2

To input this information for PARTISN, we enter in Block-IV the following:

```
MATLS= SS, CHROME 1.6676-2, NICKEL 6.566-3, IRON 6.3832-2 ;
        UO2, U235 6.7-5, U238 2.2234-2, OXYGEN 4.4602-2 ;
        PUO2, PU239 1.7761-2, PU240 4.44-3, OXYGEN 4.4402-2 ;
        NA, SODIUM 2.096-2 ;
```

Note the use of semicolons (;) to delimit the different materials indicating a separate string for each material (see the chapter "FREE FIELD INPUT REFERENCE" starting on page 5-1). Also note that the use of commas is optional. Blanks also serve as delimiters.

ASSIGNING MATERIALS TO ZONES

The macroscopic cross sections for the zones in the physical problem being analyzed are created from the material cross sections by assigning materials to zones with appropriate material concentrations, volume fractions, or densities, as desired. This assignment is accomplished either by means of the ASSIGN array card-image input in Block-IV or by means of a pre-existing code-dependent binary interface file ASGMAT.

As an example of the material assignments to zones, suppose the following materials have been created: Stainless Steel (SS), Coolant (NA), U-238 Oxide (U8O2), U-235 Oxide (U5O2), and PU-239 Oxide (PU9O2). It is desired to assign these three materials to create the correct macroscopic zone sections for the three zones named CORE, BLKT, and REFL whose compositions are as follows:

Table 7.2 Composition of Zones

Zone	Material	Material Volume Fraction
CORE	SS	0.25
	NA	0.40
	U8O2	0.20
	PU9O2	0.15
BLKT	SS	0.25
	NA	0.40
	U8O2	0.349
	U5O2	0.001
REFL	SS	0.030
	NA	0.70

The above specifications can be provided via the ASSIGN array of Block-IV of the input by entering the card-image input:

```
ASSIGN= CORE SS 0.25 NA 0.40 U8O2 0.20 PU9O2 0.15 ;
        BLKT SS 0.25 NA 0.40 U8O2 0.349 U5O2 0.001 ;
        REFL SS 0.3 NA 0.7
```

The card-image input for the assignment-of-materials-to-zones is written to a code-dependent, binary interface file named ASGMAT for use by both the Solver and Edit

Modules. The file description for ASGMAT is given in the chapter “FILE DESCRIPTIONS” starting on page 11-1.

If it is desired to use a previously created ASGMAT file for specification of the assignment-of-materials-to-zones, the user should:

- (i) omit the ASSIGN array specifications in the Block-IV card-image input or, alternatively, set the Block-I input parameter NOASG to unity, and
- (ii) ensure that the binary ASGMAT file exists and is available at the time of code execution.

ALTERNATIVE FORMS OF MIXING

This section details the use of the MATSPEC and ATWT arrays.

Historically, PARTISN has been oriented toward specifying materials by requiring input that gives the atom densities of isotopes comprising the material. This is typical of reactor-oriented computer codes. There is a nuclear analysis community, however, that specifies mixes by identifying materials by the density of the material together with the isotopes and their atomic or weight fractions in that material.

PARTISN will accept the latter type of material specification in addition to the traditional atom density type of specification. A description follows.

Using Atomic Fractions or Weight Fractions (MATSPEC)

In Block-IV of the PARTISN input (the mixing specifications) is the parameter MATSPEC. There are three allowable input values for MATSPEC:

- a. MATSPEC=ATDENS tells the code you are using the traditional atom density style of input for mixing specifications as described on page 7-9. This value is the DEFAULT.
- b. MATSPEC=ATFRAC tells the code you are using the type of mixing that specifies the density (gm/cc) of each material together with the isotopes (nuclides) and their atomic fractions in each material.
- c. MATSPEC=WTFRAC tells the code you are using the type of mixing that specifies the density (gm/cc) of each material together with the isotopes (nuclides) and their weight fractions in each material.

The MATSPEC parameter can be entered as a vector parameter with up to MT entries so that different materials can be specified with different types of mixing specifications. (Recall that MT is the number of materials to be created, as specified in Block-I of the input.) If the number of entries is less than MT, the last entry will be applied to all remaining and unspecified entries.

Following is a description of how to input information in each of the above styles.

Note: In the following, mat_m denotes the name (or identifier) of material m. It is usually a character name of the user's choice, e.g., fuel, Tu, steel, etc., for ease of reading.

$iso_{i,m}$ denotes the name (or identifier) of an isotope that exists in material m . This must be one of the names from the cross-section library. It is the i -th isotope in the specific list for material m and may or may not be the i -th isotope on the cross-section library. $iso_{i,m}$ is usually the ZAIID identifier* if one is using a MENDF cross-section library.

$af_{i,m}$ denotes the atomic fraction of isotope i that exists in material m .

$wf_{i,m}$ denotes the weight fraction of isotope i that exists in material m .

$zonid_j$ denotes the name (or identifier) of zone j . It is usually a hollerith name, of the user's choice, e.g. core, shell, driver, etc., for ease of reading and identifying.

ρ_m denotes the density [gm/cc] of material m as it exists in a zone.

- MATSPEC= ATDENS

If one makes the entry MATSPEC=ATDENS or one omits the MATSPEC= entry, the mixing is done with atom densities as described earlier in this chapter.

- MATSPEC= ATFRAC

(Step 1) One defines materials, each with a density of 1 gm/cc, by specifying (using the MATLS= input array of Block-IV) the name (or identifier) of the material followed by the isotope names (or identifiers) paired with the atomic fractions of those isotopes which define the material. This is done by inputting

MATLS= $mat_1 iso_{1,1} af_{1,1} iso_{2,1} af_{2,1} \dots ; mat_2 iso_{1,2} af_{1,2} iso_{2,2} af_{2,2} \dots ;$ etc.

NOTE: Each material's specification must be separated from the next by a semicolon!

(Step 2) Then one specifies how the above materials will be mixed and at what density into each zone. This is done by inputting

ASSIGN= $zonid_1 mat_i \rho_i \dots ; zonid_2 mat_k \rho_k mat_l \rho_l \dots ;$ etc.

NOTE: Each zone's specification must be separated from the next by a semicolon!

* The user may get a list of the available ZAIID's by making a preliminary run that only includes the Block-I and Block-III input. A listing of the ZAIID's will be found in the printed output from that run.

• MATSPEC= WTFRAC

(Step 1) One defines materials, each with a density of 1 gm/cc, by specifying (using the MATLS= input array of Block-IV) the name (or identifier) of the material followed by the isotope names (or identifiers) paired with the weight fractions of those isotopes which define the material. This is done by inputting

MATLS= *mat*₁ *iso*_{1,1} *wf*_{1,1} *iso*_{2,1} *wf*_{2,1} ... ; *mat*₂ *iso*_{1,2} *wf*_{1,2} *iso*_{2,2} *wf*_{2,2} ... ; etc.

NOTE: Each material's specification must be separated from the next by a semicolon!

(Step 2) Then one specifies how the above materials will be mixed and at what density into each zone. This is done by inputting

ASSIGN= *zonid*₁ *mat*_i ρ_i ... ; *zonid*₂ *mat*_k ρ_k *mat*_l ρ_l ... ; etc.

NOTE: Each zone's specification must be separated from the next by a semicolon!

To illustrate the use of these arrays, we repeat the example of page 7-9 with the following:

MATSPEC= WTFRAC ATFRAC

MATLS=SS, IRON 0.74, CHROME0.18, NICKEL0.08;
 UO2, U235 0.003 U238 0.997 OXYGEN2.0;
 PUO2,PU2390.8 PU240 0.2 OXYGEN2.0;
 NA, SODIUM1.0;

ASSIGN=STEEL SS 8.0;
 UFUEL UO210.0;
 PFUEL PUO210.0;
 COOLNA0.0;

Note that the density of the materials are put in the ASSIGN array whereas the respective fractions defined by the MATSPEC array are in the MATLS card. Note also that we rely on the word, ATFRAC, being repeated to fill the third and fourth positions of the MATSPEC array.

Providing Atomic Weights to the Code (the ATWT Array)

When using MATSPEC=ATFRAC or MATSPEC=WTFRAC, the code must have the atomic weights of the isotopes.

If the cross-section library is not a MENDF, MENDFG, or a BXSLIB file created with atomic weights on it, the atomic weights must be supplied in the Block-IV input. This is done as follows:

In Block-IV enter

$$\text{ATWT} = \text{iso}_1 \text{ atwt}_1 \text{ iso}_2 \text{ atwt}_2 \dots \text{iso}_n \text{ atwt}_n$$

where $n \leq \text{NISO}$, iso_i is the isotope name (identifier) for an isotope on the cross-section library, and atwt_i is its atomic weight. It is the i -th isotope in this specific list of atomic weights and may or may not be the i -th isotope on the cross-section library.

If using LIB=MENDF for neutrons or LIB=MENDFG for gammas in Block-III of the input, the atomic weights are automatically provided and nothing more need be done by the user.

NOTE: Whenever the atomic weights are provided, either in the card-image input or from a MENDF or MENDFG file, the BXSLIB file that the code creates will contain the atomic weight data automatically. By saving this BXSLIB file (see the SAVBXS parameter in Block-III) and using it as the cross-section library file in subsequent runs, there will be no need to re-enter the atomic weights in the card-image input.

INTERFACE FILES USED IN MIXING

1. Material Mixing and the Creation of Interface Files.

In the material mixing operation in the Input Module of PARTISN, the following four binary interface files are produced: MACRXS, SNXEDT, NDXSRF, and ZNATDN. These, and only these, files are used by subsequent portions of the code; the basic isotope cross-section library is “forgotten” once these four files are created.

The MACRXS code-dependent binary interface file is described in the chapter “FILE DESCRIPTIONS” starting on page 11-1 and contains material cross sections in energy-group order. The MACRXS file is the only cross-section file available to the Solver Module. If a large isotope-ordered, basic cross-section library is used, the mixing and group-ordering process used in creating the MACRXS file can be quite time-consuming. If several calculations are to be performed, for example, parametric studies on a particular nuclear system, it is advantageous to create a basic MACRXS material file one time only and save this file for use in subsequent runs involving the Solver Module. By use of the assignment-of-materials-to-zones specification in Block-IV, a single set of materials, that is, a single MACRXS file can be used for calculating numerous different problems in which the problem zone compositions consist of different proportions of materials. See “ASSIGNING MATERIALS TO ZONES” on page 7-11. The manner in which the code is instructed to use an existing MACRXS file is described below on page 7-18.

The SNXEDT code-dependent, binary interface file produced by the Input Module contains group-ordered cross-section data for use by the Edit Module. Contained in the file are the principal cross sections and edit position data for all isotopes on the basic input cross-section library. Scattering, or transfer, matrices are not included on the SNXEDT file. This file is used directly by the Edit Module for providing microscopic and constituent edits described in the chapter “RUNNING THE EDIT MODULE” starting on page 4-1. The SNXEDT file description is given in the chapter “FILE DESCRIPTIONS” on page 11-80.

The NDXSRF and ZNATDN standard interface files are used by the Edit Module together with the SNXEDT file to mix the isotopes into the materials used by the Solver Module. The Edit Module uses these materials in providing the macroscopic (or material) edits described in the edit section of any of the User’s Guides. It is again noted that in using the NDXSRF and ZNATDN files, the term “zone” in the file descriptions in Ref. 1 and Ref. 2 must be replaced with the word “material” to be consistent with our terminology.

As with the MACRXS file discussed above, it is frequently advantageous to save the SNXEDT, NDXSRF, and ZNATDN files created in one run for use in subsequent runs, if possible. This procedure eliminates the need to continually repeat the often time-consuming process of re-creating the group-ordered code-dependent SNXEDT file. Parametric studies on variations of material compositions in the zones of the physical problem can be accomplished simply by changing the assignment-of-materials-to-zones specifications in Block-V. See “ASSIGNING MATERIALS TO ZONES” on page 7-11.

The manner in which the code is instructed to use existing SNXEDT, NDXSRF, and ZNATDN files is described below. It should be noted that the use of an SNXEDT file by the Edit Module is usually accompanied by the use of the associated NDXSRF and ZNATDN files, and it is wise to treat these three files as a single triumvirate.

2. Using Existing MACRXS, SNXEDT, NDXSRF, ZNATDN Interface Files.

If an existing pair of NDXSRF and ZNATDN standard interface files is to be used to specify the material mixing instructions in conjunction with a basic isotope cross-section library, the user should

- (i) omit the specification of the MATLS array in Block-IV card-image input or, alternatively, set the Block-I input parameter NOMIX to unity, and
- (ii) ensure that the NDXSRF and ZNATDN binary files exist and are available at the time of execution.

If an existing quartet of MACRXS, SNXEDT, NDXSRF, and ZNATDN binary interface files is to be used, the user should

- (i) omit Block-III and the MATLS array in Block-IV in the card-image input or, alternatively, set LIB=MACRXS in the Block-III input or, alternatively, set the Block-I input parameters NOMIX and NOMACR both in unity, and
- (ii) ensure that the MACRXS, SNXEDT, NDXSRF, and ZNATDN binary files exist and are available at the time of execution. Note: only the MACRXS, NDXSRF, and ZNATDN files are needed for execution of the Solver Module, and only the SNXEDT, NDXSRF, and ZNATDN files are needed for execution of the Edit Module.

REFERENCES

1. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
2. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physics Codes, Version III," Los Alamos Scientific Laboratory report LA-5486-MS (February 1974).

MIXING CHAPTER INDEX

A

ASSIGN

Example of use 7-11

F

Files

Used in mixing 7-17

M

MATLS

Example of use 7-9

Mixing

ASSIGN input array example..... 7-11

MATLS input array example..... 7-9

Terminology 7-7

Tutorial 7-1

Using atomic fractions..... 7-13

Using weight fractions..... 7-13

While assigning materials to zones 7-11

Z

ZAID isotope name..... 7-14

METHODS MANUAL

Transport Methods Group, CCS-4
Los Alamos National Laboratory

8

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	8-3
LIST OF FIGURES	8-5
LIST OF TABLES.....	8-7
INTRODUCTION	8-9
The First Order Form of the Boltzmann Transport Equation.....	8-9
MULTIGROUP, DISCRETE ORDINATES TRANSPORT THEORY.....	8-11
Multigroup Approximation	8-11
Discrete Ordinates Approximation.....	8-12
Iteration Procedure and Diffusion Synthetic Acceleration.....	8-14
ONE-DIMENSIONAL METHODS.....	8-19
Geometrical Symmetries Treated in One Dimension.....	8-19
Particular Forms of the Divergence Operator	8-21
Spherical Harmonics Expansion of the Scattering Source	8-23
Spherical Harmonics Expansion of the Inhomogeneous Source.....	8-28
Discrete-Ordinates Equations in One Dimension.....	8-29
Discretization of the Spatial Variable	8-35
TWO-DIMENSIONAL METHODS.....	8-37
Some Angular Details in Two Dimensions	8-37
Transport Operator in Two-Dimensional Symmetries	8-38
Planar X-Y Symmetry	8-39
Planar R- Θ Symmetry	8-39
Cylindrical R-Z Symmetry	8-39
Spatially Discretized Two-Dimensional Transport Equation.....	8-39
THREE-DIMENSIONAL METHODS.....	8-43
REFERENCES	8-55
METHODS CHAPTER INDEX	8-46

LIST OF FIGURES

Figure 8.1: Coordinates in plane geometry	8-19
Figure 8.2: Coordinates in cylindrical geometry	8-20
Figure 8.3: Coordinates in spherical geometry	8-20
Figure 8.4: Ordering of S_6 directions in plane and spherical geometries	8-30
Figure 8.5: Ordering of S_4 directions in two-angle plane geometry	8-31
Figure 8.6: Ordering of S_6 directions in cylindrical geometry	8-32

LIST OF TABLES

Table 18.1: Forms of the Divergence Operator	8-22
Table 18.2: Number of Spherical Harmonics, N , as a Function of Order	8-25
Table 18.3: Spherical Harmonics, for Different Geometries	8-26
Table 18.4: Number of Quadrature Points, M as a Function of S_n Order, N	8-29
Table 18.5: Spherical Harmonics in Two Dimensions.....	8-37
Table 18.6: Number of Angles Per Octant in Two Dimensions	8-38

INTRODUCTION

This chapter provides the development of the multigroup, discrete-ordinates, form of the time-independent Boltzmann transport equation.^{1,2} This development is followed by a brief description of the iterative procedure used to solve the transport equation employing the diffusion synthetic acceleration (DSA) scheme to accelerate the iterations³. The specifics of the discrete ordinates method is provided for the PARTISN solver module. We also include some of the details on spatial discretization as is appropriate for each geometry.

The First Order Form of the Boltzmann Transport Equation

The time-independent inhomogeneous Boltzmann transport equation may be written as:

$$\begin{aligned}
 & \mathbf{v} \cdot \underline{\underline{\Omega}} \nabla \Psi(r, E, \underline{\underline{\Omega}}) + \sigma(r, E) \Psi(r, E, \underline{\underline{\Omega}}) \\
 &= \iint dE' d\Omega' \sigma_s(r, E' \rightarrow E, \underline{\underline{\Omega}} \cdot \underline{\underline{\Omega}}') \Psi(r, E', \underline{\underline{\Omega}}') \\
 &+ \frac{1}{k_{eff}} \iint dE' d\Omega' \chi(r, E' \rightarrow E) \nu \sigma_f(r, E') \Psi(r, E', \underline{\underline{\Omega}}') \\
 &+ O(r, E, \underline{\underline{\Omega}}),
 \end{aligned} \tag{1}$$

where $\Psi(r, E, \underline{\underline{\Omega}})$ is the particle flux (particle number density times the particle speed defined such that $\Psi(r, E, \underline{\underline{\Omega}}) dE \, dr \, d\Omega$ is the flux of particles in the energy range dE about E , in the volume element dr about r , with directions of motion in the solid angle element $d\Omega$ about $\underline{\underline{\Omega}}$. Similarly, $Q(r, E, \underline{\underline{\Omega}}) dE \, dr \, d\Omega$ is the rate at which particles are produced in the same element of phase space from sources that are independent of the flux Ψ . The macroscopic total cross section is σ . Assuming isotropic media, the macroscopic scattering transfer cross section, from energy E' to energy E through a scattering angle $\underline{\underline{\Omega}} \cdot \underline{\underline{\Omega}}'$, is σ_s . And the macroscopic fission cross section is σ_f . All of the quantities may be spatially dependent. The number of particles emitted isotropically $\left(\frac{1}{4\pi}\right)$ per fission is ν , and the fraction of these particles appearing in energy dE about E from fissions in dE' about E' is $\chi(r, E' \rightarrow E)$. If Q is not zero in Eq. (1), then k_{eff} is set to 1; if Q is zero, then the problem is an eigenvalue problem and $1/k_{eff}$ is the eigenvalue.

The remainder of this chapter is devoted to the consideration of the discretization of this equation as employed in each of the solver modules. The energy and angular variable discretization is common to all the modules and hence is developed first. The spatial discretization is explained separately for each of the modules concerned.

MULTIGROUP, DISCRETE ORDINATES TRANSPORT THEORY

We begin the discretization of the neutral particle, Boltzmann transport equation by considering the phase space variables of energy and angle. This is the multigroup in energy and discrete ordinates in angle approximation which is common to all of PARTISN

Multigroup Approximation

In the energy variable, the energy domain is partitioned into G intervals of width $\Delta E_g = E_{g-1/2} - E_{g+1/2}$. By convention, the highest energy is at $E_{1/2}$ and hence the index g increases as energy decreases (since the normal transport of particles in energy is from high to low energy as they collide with nuclei in the medium). Thus if we integrate Eq. (1) over each energy interval, we obtain the following discretized equation:

$$\underline{\Omega} \cdot \nabla \psi_g(\underline{r}, \underline{\Omega}) + \sigma_{t,g}(\underline{r}) \psi_g(\underline{r}, \underline{\Omega}) = \frac{\chi_g(\underline{r})}{k_{eff}} \Phi(\underline{r}) + Q_g(\underline{r}, \underline{\Omega}) + \sum_{g'=1}^G \int \sigma_{s,g' \rightarrow g}(\underline{r}, \mu_0) \psi_{g'}(\underline{r}, \underline{\Omega}') d\underline{\Omega}' \quad (2)$$

$$\sum_{g'=1}^G \int \sigma_{s,g' \rightarrow g}(\underline{r}, \mu_0) \psi_{g'}(\underline{r}, \underline{\Omega}') d\underline{\Omega}'$$

$$g = 1, \dots, G$$

where, $\mu_0 = \underline{\Omega} \cdot \underline{\Omega}'$,

$$\psi_g(\underline{r}, \underline{\Omega}) = \int_{\Delta E_g} \psi(\underline{r}, \underline{\Omega}, E) dE,$$

$$\sigma_{t,g}(\underline{r}) = \left(\int_{\Delta E_g} \sigma_t(\underline{r}, E) \psi(\underline{r}, \underline{\Omega}, E) dE \right) / \psi_g(\underline{r}, \underline{\Omega}),$$

$$\sigma_{s, g' \rightarrow g}(\underline{r}, \underline{\mu}_0) = \left(\int_{\Delta E_g} \int_{\Delta E_{g'}} \sigma(\underline{r}, E' \rightarrow E, \underline{\mu}_0) \psi(\underline{r}, \underline{\Omega}, E') dE dE' \right) / \psi_{g'}(\underline{r}, \underline{\Omega}),$$

$$Q_g(\underline{r}, \underline{\Omega}) = \int_{\Delta E_g} Q(\underline{r}, \underline{\Omega}, E) dE,$$

$$\chi_g(\underline{r}) = \int_{\Delta E_g} \chi(\underline{r}, E) dE,$$

$$\Phi(\underline{r}) = \sum_{g=1}^G \left(\int_{\Delta E_g} v \sigma_f(\underline{r}, E) \phi(\underline{r}, E) dE \right) / (\phi_g(\underline{r})),$$

$$\phi_g(\underline{r}) = \int \psi_g(\underline{r}, \underline{\Omega}) d\underline{\Omega}.$$

Note that the definition of the multigroup cross sections is a formal one since one would need to know the solution of the transport problem in order to evaluate them. In practice, the multigroup cross sections are supplied from some cross-section processing code which predefines the energy intervals and the weighting functions. We also note that much of the physics of the problem is contained in the cross-section set and hence this aspect of the solution process of the transport problem should not be minimized. That is, a careful selection of the cross-section set should be made in order to ensure a physically accurate solution. For few groups, this accuracy is heavily dependent upon the weighting function; as the number of groups increases, this becomes less so assuming a proper treatment of the resonance region.

Discrete Ordinates Approximation

The next discretization involves the angular variable. In the method of discrete ordinates, the angle is discretized over the unit sphere in a prescribed manner. The choice of the discretization seeks to satisfy the following conditions:

- (1) physical symmetries are preserved upon discretization,
- (2) spherical harmonics moments are well approximated in order to provide an accurate representation of the source terms, and
- (3) derivatives with respect to the angle coordinates that come from the streaming operator are simply approximated.

Since in multidimensional cases not all of the above conditions can be simultaneously satisfied, compromises are made in defining discrete ordinates sets. To understand these points more completely see Ref. 4. Basically each discrete ordinate is depicted as an angular direction vector, $\underline{\Omega}_m$, with an associated area on the unit sphere, w_m , where

$m=1, \dots, M$, M being the number of discrete ordinates. This M is derived from the S_n order specified in Block-I and depends upon the set arrangement chosen and the number of spatial dimensions. All of the discrete ordinates sets used in the solver modules satisfy certain, fundamental conditions including $\sum_{m=1}^M w_m = 4\pi^*$ for conservation. We also impose the requirements that:

$$\sum_{m=1}^M w_m \underline{\Omega}_m = 0 \text{ for symmetry, and}$$

$$\sum_{m=1}^M w_m \mu_m^2 = \sum_{m=1}^M w_m \eta_m^2 = \sum_{m=1}^M w_m \xi_m^2 = \frac{1}{3} \text{ for the diffusion limit.}$$

where the components of $\underline{\Omega}_m$ are: $\underline{\Omega}_m = \mu_m \underline{i} + \eta_m \underline{j} + \xi_m \underline{k}$.

To obtain the discrete ordinates balance equation, we integrate Eq (2) over w_m (in this case we assume isotropic scattering for simplicity):

$$[\underline{\Omega} \cdot \nabla \psi_g(\underline{r}, \underline{\Omega})]_m + \sigma_{t,g}(\underline{r}) \psi_{g,m}(\underline{r}) = \frac{\chi_g(\underline{r})}{k_{eff}} \Phi(\underline{r}) + Q_{g,m}(\underline{r}) + \sum_{g'=1}^G \sigma_{s0,g' \rightarrow g}(\underline{r}) \phi_{g'}(\underline{r}) \quad (3)$$

where,

$$\psi_{g,m}(\underline{r}) = \int_{w_m} \psi(\underline{r}, \underline{\Omega}) d\underline{\Omega},$$

$$\phi_g(\underline{r}) = \sum_{m=1}^M w_m \psi_{g,m}(\underline{r}),$$

$$Q_{g,m}(\underline{r}) = \sum_{l=0}^L \sum_{a=-l}^l (2l+1) Y_l^a(\underline{\Omega}_m) Q_{g,l}^q(\underline{r}),$$

$$Q_{g,l}^q(\underline{r}) = \int Y_l^{*q}(\underline{\Omega}) Q_g(\underline{r}, \underline{\Omega}) d\underline{\Omega},$$

*In this code we renormalize the weights such that $\sum_{m=1}^M w_m = 1$.

and the $Y_l^q(\underline{\Omega})$ are the spherical harmonic functions.

Note that the streaming operator is only discretized symbolically; the specific form of this operator depends upon the geometric symmetry chosen and to some extent the spatial discretization method. We also note that we have expanded the source in spherical harmonics, the form in which the code expects the angular dependence of the source to be represented if it is a volume source. Discussion of the representation of an anisotropic scattering source is presented in the section ‘‘Spherical Harmonics Expansion of the Scattering Source’’ on page 8-23 below.

Iteration Procedure and Diffusion Synthetic Acceleration

In solving the transport equation numerically, an iterative procedure is used. This procedure involves two levels of iteration referred to as inner and outer iterations. The acceleration of these iterations is of crucial importance to transport codes in order to reduce the computation time involved. The PARTISN Solver Module uses the diffusion synthetic acceleration (DSA) method developed by Alcouffe,³ an extremely effective method for accelerating the convergence of the iterations.

To display the iterative procedure and the application of the diffusion synthetic acceleration method, consider first the inner iteration equation for energy group g and inner iteration l . Isotropic scatter is assumed only for simplicity. The basic inner iteration equation is written

$$\underline{\Omega} \cdot \nabla \tilde{\Psi}_g^l(r, \underline{\Omega}) + \sigma_g(r) \tilde{\Psi}_g^l(r, \underline{\Omega}) = \sigma_{s,g \rightarrow g}(r) \phi_g^{l-1}(r) + QQ_g(r). \quad (4)$$

In Eq. (4), $\tilde{\Psi}_g^l(r, \underline{\Omega})$ is the angular flux for group g at the l^{th} inner iteration using a scalar flux $\phi_g^{l-1}(r)$ assumed known from the previous inner iteration. QQ_g is the group source which remains unchanged for the group throughout the performance of inner iterations. This group source contains scattering and fission contributions to the group together with any inhomogeneous source. The source is computed using the multigroup scalar fluxes and moments from the previous outer iteration. In the diffusion synthetic method, a corrected diffusion equation is used to determine the scalar flux ϕ_g^l needed for the next iteration. In fact, there are three separate schemes for writing the corrected diffusion equation to be used: the source correction scheme, the diffusion coefficient correction scheme, and the removal correction scheme. For the source correction scheme we write the corrected diffusion equation as

$$-\nabla \cdot D_g(r) \nabla \phi_g^l(r) + \sigma_{R,g}(r) \phi_g^l(r) = QQ_g(r) - R_g^l(r), \quad (5)$$

where

$$D_g(r) = \frac{1}{3\sigma_{tr,g}}(r), \quad \sigma_{R,g}(r) = \sigma_g(r) - \sigma_{s,g \rightarrow g}(r)$$

and the correction term is

$$R_g^l(r) = \nabla \cdot \tilde{J}_g^l(r) + \nabla \cdot D_g(r) \nabla \tilde{\phi}_g^l(r). \quad (6)$$

In Eq. (6),

$$\tilde{\phi}_g^l(r) = \int d\Omega \tilde{\psi}_g^l(r, \underline{\Omega}), \quad \tilde{J}_g^l(r) = \int d\Omega \underline{\Omega} \tilde{\psi}_g^l(r, \underline{\Omega}). \quad (7)$$

Note that a tilde is used to indicate quantities calculated using the transport angular flux, $\tilde{\psi}_g^l$, while the scalar flux calculated from the corrected diffusion equation is without the tilde.

The source correction scheme for the inner iteration proceeds as follows: using ϕ_g^{l-1} , known from the previous iteration, Eq. (4) is solved for $\tilde{\psi}_g^l$. This involves one sweep through the space-angle mesh. The correction term, R_g^l , is then calculated using Eqs. (6) and (7) and, in turn, used in Eq. (5) to calculate ϕ_g^l to complete one cycle or one inner iteration. The steps are repeated until suitable convergence is achieved. Note that for the first inner iteration for a group, a logical first guess for the scalar flux is obtained by solving Eq. (5) with R_g set to zero for $l=0$.

It is easy to show that if the iteration converges, it converges to the transport equation solution. Namely, drop all l superscripts and set the transport scalar flux to the corrected diffusion scalar flux, $\tilde{\phi}_g = \phi_g$. Then substituting Eq. (6) into Eq. (5) yields

$$\nabla \cdot \tilde{J}_g(r) + \sigma_{R,g}(r) \tilde{\phi}_g(r) = Q_g,$$

which is the converged transport balance equation obtained also by integrating Eq. (4) over all $\underline{\Omega}$.

After performing the inner iterations for each group using Eqs. (4) through (7) to obtain the group converged correction terms $R_g^k(r)$, a second level of iteration is needed to update the multigroup sources. A multigroup, corrected diffusion equation is constructed for this purpose which has the following form:

$$\begin{aligned}
-\nabla \cdot D_g(r) \nabla \phi_g^{k+1}(r) + \sigma_{R,g}(r) \phi_g^{k+1}(r) &= Q_g(r) - R_g^k(r) \\
&+ \chi_g \sum_{g'=1}^G \nu \sigma_{f,g'}(r) \phi_{g'}^{k+1}(r) + \sum_{g' \neq g} \sigma_{s,g' \rightarrow g}(r) \phi_{g'}^{k+1}(r)
\end{aligned} \tag{8}$$

Equation (8) is necessary if there are fissions or upscattering processes in the problem. The solution of this multigroup, diffusion equation in itself involves an iteration procedure which we term sub-outer iteration. In using this second level of iteration, we are effectively replacing transport outer iterations by the diffusion sub-outers.

The source correction scheme outlined above using the diffusion synthetic method is an effective scheme for inhomogeneous source problems. For eigenvalue problems, Eq. (8) must be homogeneous, and it is necessary to define a different scheme for the diffusion synthetic method. The diffusion coefficient correction scheme is one such scheme. In this scheme we redefine the corrected diffusion coefficient $\underline{\underline{D}}_g(r)$ as

$$\underline{\underline{D}}_g(r) = \frac{-\tilde{J}_g(r)}{\nabla \tilde{\phi}_g(r)}, \tag{9}$$

so that $R_g(r) = 0$ for all r and g . Then with $Q_g(r) = 0$, the inner iteration diffusion equation becomes

$$-\nabla \underline{\underline{D}}_g(r) \cdot \nabla \phi_g^l(r) + \sigma_{R,g}(r) \phi_g^l(r) = Q_g(r), \tag{10}$$

and the multigroup (outer iteration) diffusion equation becomes

$$\begin{aligned}
-\nabla \underline{\underline{D}}_g(r) \cdot \nabla \phi_g^{k+1}(r) + \sigma_{R,g}(r) \phi_g^{k+1}(r) &= \frac{\chi_g}{k_{eff}} \sum_{g'=1}^G \nu \sigma_{f,g'}(r) \phi_{g'}^{k+1}(r) \\
&+ \sum_{g' \neq g} \sigma_{s,g' \rightarrow g}(r) \phi_{g'}^{k+1}(r),
\end{aligned} \tag{11}$$

where k_{eff} is the multiplication factor for the system. The same iteration procedure is used for this diffusion coefficient correction scheme as for the source correction scheme.

For eigenvalue problems, the diffusion correction scheme has been found to accelerate the iterations as readily as the source correction scheme for inhomogeneous source problems. In fact in PARTISN, the diffusion coefficient correction scheme is used for inhomogeneous source problems in which fission and/or upscatter is present while the source

correction scheme is used only for inhomogeneous source problems with downscatter and no fission.

One disadvantage to the diffusion coefficient correction scheme is that infinite and negative diffusion coefficients are possible (see Eq. (9)). If this occurs, Eq. (10) cannot be solved using current techniques. To overcome this difficulty, the removal correction scheme is employed. A corrected removal cross section is defined as

$$\tilde{\sigma}_{R,g}^l(r) \equiv \sigma_{R,g}(r) + \frac{R_g^l(r)}{\tilde{\phi}_g^l(r)}, \quad (12)$$

where $R_g^l(r)$ is defined by Eq. (6). With this modification, Eq. (5) becomes

$$-\nabla \cdot D_g(r) \nabla \phi_g^l(r) + \tilde{\sigma}_{R,g}^{l-1}(r) \phi_g^l(r) = QQ_g(r)$$

and Eq. (8) becomes

$$\begin{aligned} & -\nabla \cdot D_g(r) \nabla \phi_g^{k+1}(r) + \tilde{\sigma}_{R,g}(r) \phi_g^{k+1}(r) \\ & = \frac{\chi_g}{k_{eff}} \sum_{g'=1}^G \nu \sigma_{f,g'}(r) \phi_{g'}^{k+1} + \sum_{g' \neq g} \sigma_{s,g' \rightarrow g}(r) \phi_{g'}^{k+1}(r). \end{aligned} \quad (13)$$

The iteration procedure is entirely analogous to that for the diffusion coefficient correction scheme and again, if it converges, it converges to the transport balance equation solution. This removal correction scheme is employed in eigenvalue problems or source problems with fission and/or upscatter only when the diffusion coefficient correction scheme produces negative or infinite diffusion coefficients.

ONE-DIMENSIONAL METHODS

This section is devoted to the description of the angular and spatial discretization forms that are specific to the one-dimensional geometries treated in PARTISN. The one-dimensional symmetries offer great simplification of the transport process and allow rapid transport calculations for physical systems that can be reasonably represented as one dimensional.

Geometrical Symmetries Treated in One Dimension

The symmetries treated in one dimension are: (1) slab, (2) infinite cylinder, (3) sphere, and (4) two-angle slab. The two-angle slab case treats slab geometries where the angular dependence of an incident source (boundary source) is possibly outside the plane normal to the surface of the slab.

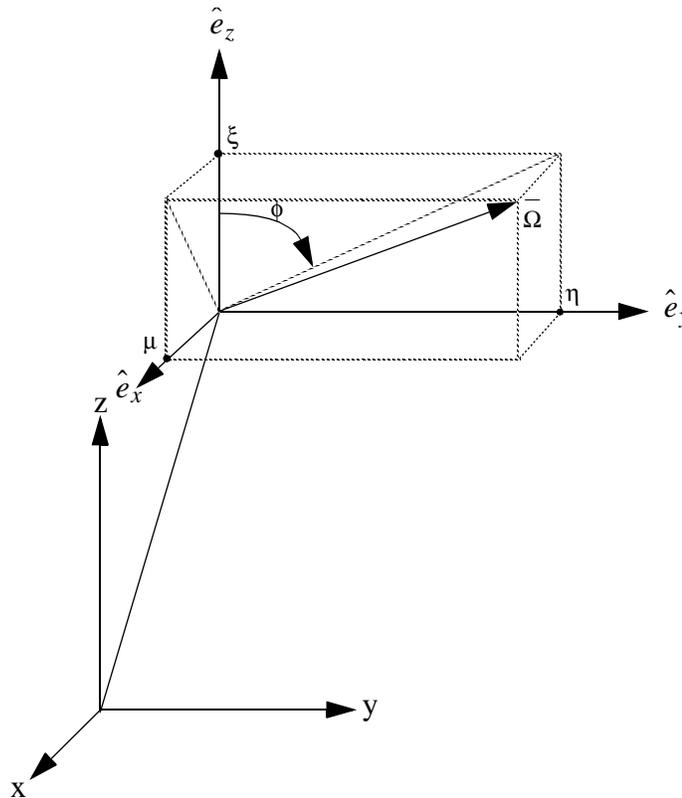


Figure 8.1 Coordinates in plane geometry

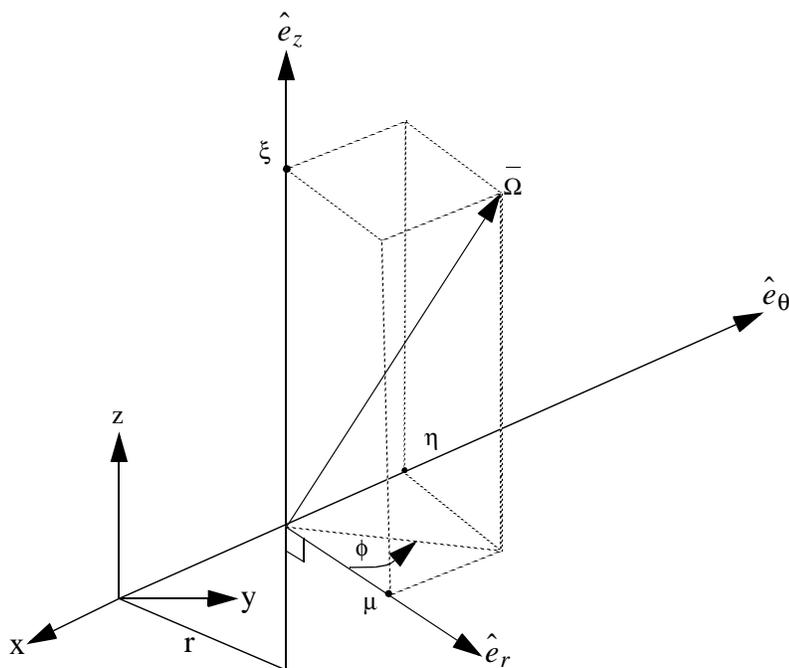


Figure 8.2 Coordinates in cylindrical geometry

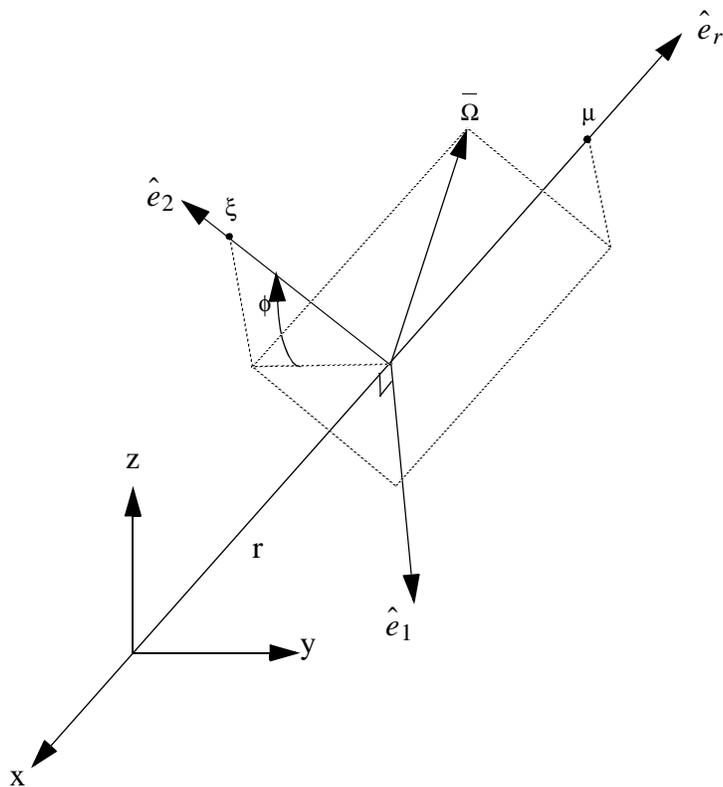


Figure 8.3 Coordinates in spherical geometry

Particular Forms of the Divergence Operator

The divergence operator in conservative form, $\nabla \cdot \underline{\Omega}\psi$ or $(\underline{\Omega} \cdot \nabla)\psi$ for the geometries treated in one-dimension is given in Table 8.1 in terms of the coordinate systems shown in Figure 8.1 through Figure 8.3.

In the standard plane geometry, the angular flux $\psi(r, E, \underline{\Omega})$ is assumed independent of the azimuthal angle ϕ so that the angular dependence is reduced to the μ interval $(-1, +1)$. PARTISN also permits the two-angle plane geometry option in which no assumptions of symmetry in angle are imposed. In this case the complete unit sphere of angular directions must be considered.

In cylindrical geometry, the angular flux is assumed symmetric in the ξ angular cosine and also symmetric about the $\mu - \xi$, (or $\phi = 0^\circ - 180^\circ$) plane. Thus, only one-fourth of the unit sphere need be considered in the angular dependence.

In spherical geometry, the angular flux is assumed symmetric in the azimuthal angle ϕ so that the angular dependence is reduced to the interval $(-1, +1)$.

Table 8.1 Forms of the Divergence Operator

Geometry	Dependence of	Definition of Variables	$\nabla \cdot \underline{\Omega}\psi$
Plane	$\psi(x, \mu)$	$\mu = \hat{e}_x \cdot \underline{\Omega}$	$\mu \frac{\partial \psi}{\partial x}$
	or $\psi(x, \mu, \phi)$	$\xi = (1 - \mu^2)^{1/2} \cos \phi$ $\eta = (1 - \mu^2)^{1/2} \sin \phi$	
Cylindrical	$\Psi(x, \mu, \eta)$	$\mu = \hat{e}_r \cdot \underline{\Omega}$ $\xi = \hat{e}_z \cdot \underline{\Omega}$ $\eta = (1 - \xi^2)^{1/2} \sin \phi$ $\mu = (1 - \xi^2)^{1/2} \cos \phi$	$\frac{\mu}{r} \frac{\partial(r\psi)}{\partial r} - \frac{1}{r} \frac{\partial(\eta\psi)}{\partial \phi}$
Spherical	$\Psi(r, \mu)$	$\mu = \hat{e}_r \cdot \underline{\Omega}$	$\frac{\mu}{r^2} \frac{\partial(r^2\psi)}{\partial r} + \frac{1}{r} \frac{\partial[(1 - \mu^2)\psi]}{\partial \mu}$

Spherical Harmonics Expansion of the Scattering Source

The scattering transfer cross section in Eq. (1) is represented by a finite Legendre polynomial expansion of order ISCT

$$\sigma_s(r, E' \rightarrow E, \underline{\Omega}' \cdot \underline{\Omega}) = \sum_{L=0}^{ISCT} \left(\frac{2L+1}{4\pi} \right) \sigma_s^L(r, E' \rightarrow E) P_L(\underline{\Omega}' \cdot \underline{\Omega}) \quad (14)$$

If this expansion is inserted into Eq. (1) and the addition theorem for spherical harmonics used to expand $P_n(\underline{\Omega}' \cdot \underline{\Omega})$, the scattering source becomes

$$SS = \int dE' \sum_{L=0}^{ISCT} \left(\frac{2L+1}{4\pi} \right) \sigma_s^L(r, E' \rightarrow E) \left\{ \begin{array}{cc} 1 & 2\pi \\ P_L(\mu) \int d\mu' \int d\phi' P_L(\mu') \psi(r, E', \mu', \phi') & \\ -1 & 0 \end{array} \right. \quad (15)$$

$$+ 2 \sum_{K=1}^L \left. \begin{array}{cc} 1 & 2\pi \\ \frac{(L-K)!}{(L+K)!} P_L^K(\mu) \int d\mu' \int d\phi' P_L^K(\mu') \cos K(\phi - \phi') \psi(r, E', \mu', \phi') & \\ \end{array} \right\}$$

where for cylindrical geometry we must replace the μ variable with ξ . Using the relation $\cos L(\phi - \phi') = \cos L\phi \cos L\phi' + \sin L\phi \sin L\phi'$, we can write Eq. (15) as

$$SS = \int dE' \sum_{L=0}^{ISCT} (2L+1) \sigma_s^L(r, E' \rightarrow E) \left\{ \begin{array}{c} P_L(\mu) \phi_L(r, E') \\ + \sum_{K=1}^L \sqrt{\frac{2(L-K)!}{(L+K)!}} \left[\phi_{C,L}^K(r, E') P_L^K(\mu) \cos K\phi + \phi_{S,L}^K(r, E') P_L^K(\mu) \sin K\phi \right] \end{array} \right\} \quad (16)$$

where we have defined the moments of the angular flux as

$$\phi_L(r, E') \equiv \frac{1}{4\pi} \int_{-1}^1 d\mu' \int_0^{2\pi} d\phi' P_L(\mu') \psi(r, E', \mu', \phi'), \quad (17a)$$

$$\phi_{C,L}^K(r, E') \equiv \frac{1}{4\pi} \int_{-1}^1 d\mu' \int_0^{2\pi} d\phi' \psi(r, E', \mu', \phi') \sqrt{\frac{2(L-K)!}{(L+K)!}} P_L^K(\mu') \cos K\phi' \quad (17b)$$

$$\phi_{S,L}^K(r, E') \equiv \frac{1}{4\pi} \int_{-1}^1 d\mu' \int_0^{2\pi} d\phi' \psi(r, E', \mu', \phi') \sqrt{\frac{2(L-K)!}{(L+K)!}} P_L^K(\mu') \sin K\phi' \quad (17c)$$

In both standard plane and spherical geometries, due to symmetry in the azimuthal angle ϕ , the flux moments $\phi_{C,L}^K$ and $\phi_{S,L}^K$ are identically zero. In cylindrical geometry (with ξ, ξ' replacing μ, μ' in Eqs. (16) and (17), the odd moments ($K+L=\text{odd}$) of $\phi_{C,L}^K$ vanish as do all the sine moments $\phi_{S,L}^K$. In the two-angle plane geometry all moments must be retained.

In all cases the scattering source, SS , can be written in the general form

$$SS = \int_E dE' \sum_{n=1}^{NM} (2L+1) \sigma_S^L(r, E' \rightarrow E) R_n(\underline{\Omega}) \check{\phi}_n(r, E'), \quad (18)$$

where NM is the total number of spherical harmonics (and flux moments) required for a given Legendre expansion order, L_o (as shown in Table 8.2), the $R_n(\underline{\Omega})$ are the spherical harmonics appropriate to the particular geometry, and the $\check{\phi}_n(r, E)$ are the angular flux moments corresponding to the $R_n(\underline{\Omega})$. $ISCT$ is the input variable for L_o , describing the expansion order of the fluxes and the scattering source. The index L in Eq. (18) is the subscript of the Legendre function P_L^K appearing in the appropriate $R_n(\underline{\Omega})$, $0 \leq L \leq ISCT$. The $R_n(\underline{\Omega})$ are listed in Table 8.3 for typical Legendre expansion orders. For each $R_n(\underline{\Omega})$ in the table, there is a corresponding flux moment defined by Eq. (17a), (17b), or (17c) as appropriate.

Table 8.2 Number of Spherical Harmonics, N , as a Function of Order L_o

L_o	Standard Plane and Spherical Geometries	Cylindrical Geometry	Two-Angle Plane Geometry
0	1	1	1
1	2	2	4
2	3	4	9
3	4	6	16
4	5	9	25
5	6	12	36

$$N = \begin{cases} L_o + 1 & \text{for standard plane and spherical geometry} \\ (L_o + 2)^2 / 4 & \text{for cylindrical geometry} \\ (L_o + 1)^2 & \text{for two-angle plane geometry} \end{cases}$$

Table 8.3 Spherical Harmonics, $R_n(\Omega)$, for Different Geometries

n	Standard Plane and Spherical Geometries P_5^a	Cylindrical Geometry P_4	Two-Angle Plane P_3
1	$\underline{P_0(\mu)}$	$\underline{P_0(\xi)}$	$\underline{P_0(\mu)}$
2	$\underline{P_1(\mu)}$	$\underline{P_1^1(\xi)\cos\phi}$	$\underline{P_1(\mu)}$
3	$\underline{P_2(\mu)}$	$\underline{P_2(\xi)}$	$\underline{P_1^1(\mu)\cos\phi}$
4	$\underline{P_3(\mu)}$	$\underline{\frac{\sqrt{3}}{6} P_2^2(\xi)\cos 2\phi}$	$\underline{P_1^1(\mu)\sin\phi}$
5	$\underline{P_4(\mu)}$	$\underline{\frac{\sqrt{6}}{6} P_3^1(\xi)\cos 2\phi}$	$\underline{P_2(\mu)}$
6	$\underline{P_5(\mu)}$	$\underline{\frac{\sqrt{10}}{60} P_3^3(\xi)\cos 3\phi}$	$\underline{\frac{\sqrt{3}}{3} P_2^1(\mu)\cos\phi}$
7		$\underline{P_4(\xi)}$	$\underline{\frac{\sqrt{3}}{3} P_2^1(\mu)\sin\phi}$
8		$\underline{\frac{\sqrt{5}}{30} P_4^2(\xi)\cos 2\phi}$	$\underline{\frac{\sqrt{3}}{6} P_2^2(\mu)\cos 2\phi}$
9		$\underline{\frac{\sqrt{35}}{840} P_4^4(\xi)\cos 4\phi}$	$\underline{\frac{\sqrt{3}}{6} P_2^2(\mu)\sin 2\phi}$
10			$\underline{P_3(\mu)}$
11			$\underline{\frac{\sqrt{6}}{6} P_3^1(\mu)\cos\phi}$
12			$\underline{\frac{\sqrt{6}}{6} P_3^1(\mu)\sin\phi}$
13			$\underline{\frac{\sqrt{15}}{30} P_3^2(\mu)\cos 2\phi}$
14			$\underline{\frac{\sqrt{15}}{30} P_3^2(\mu)\sin 2\phi}$
15			$\underline{\frac{\sqrt{10}}{60} P_3^3(\mu)\cos 3\phi}$
16			$\underline{\frac{\sqrt{10}}{60} P_3^3(\mu)\sin 3\phi}$

a. P_l denotes L-th order Legendre Expansion.

Spherical Harmonics Expansion of the Inhomogeneous Source

In a manner similar to that used for the scattering source, the inhomogeneous source $Q(r, E, \underline{\Omega})$ can be represented as a finite expansion using the spherical harmonics $R_n(\underline{\Omega})$ defined in Table 8.3. First, the inhomogeneous source moments are defined for a Legendre expansion order IQAN:

$$Q_L(r, E) \equiv \frac{1}{4\pi} \int_{-1}^1 du \int_0^{2\pi} d\phi Q(r, E, \underline{\Omega}) P_L(\mu), \quad L = 0, \dots, IQAN, \quad (19a)$$

$$Q_{C,L}^K(r, E) \equiv \frac{1}{4\pi} \int_{-1}^1 d\mu \int_0^{2\pi} d\phi Q(r, E, \underline{\Omega}) P_L^K(\mu) \cos K\phi, \quad (19b)$$

$$L = 0, \dots, IQAN \quad K = 1, \dots, L,$$

$$Q_{S,L}^K(r, E) \equiv \frac{1}{4\pi} \int_{-1}^1 d\mu \int_0^{2\pi} d\phi Q(r, E, \underline{\Omega}) P_L^K(\mu) \sin K\phi \quad (19c)$$

The inhomogeneous source is represented in the general spherical harmonic expansion

$$Q(r, E, \underline{\Omega}) = \sum_{n=1}^{NMQ} (2L+1) R_n(\underline{\Omega}) \tilde{Q}_n(r, E), \quad (20)$$

where NMQ is the total number of spherical harmonics (and source moments) required for a given Legendre expansion order, L_o , as shown in Table 8.2, the $R_n(\underline{\Omega})$ are the spherical harmonics appropriate to the geometry being used, and the $\tilde{Q}_n(r, E)$ are the angular source moments corresponding the $R_n(\underline{\Omega})$. IQAN is determined from the number of spherical harmonics, NMQ, in the input. The index L is the subscript of the Legendre function P_L^K appearing in the appropriate $R_n(\underline{\Omega})$, $0 \leq L \leq IQAN$. The $R_n(\underline{\Omega})$

are listed in Table 8.3 for typical Legendre expansion orders. For each of these $R_n(\underline{\Omega})$ is a corresponding source moment defined by Eqs. (19a), (19b), or (19c), as appropriate.

Discrete-Ordinates Equations in One Dimension

The number of discrete ordinates or angles in the various one-dimensional symmetries depend upon the symmetry and the S_n order desired. In Table 12.4 is given the number of angles used in each case.

Table 8.4 Number of Quadrature Points, M as a Function of S_n Order, N

N	Standard Plane Geometry	Two-Angle Plane Geometry	Cylindrical Geometry	Spherical Geometry
2	2	8	2	2
4	4	24	6	4
6	6	48	12	6
8	8	80	20	8
12	12	168	42	12
16	16	288	72	16
N	N	$N \cdot (N + 2)$	$\frac{N \cdot (N + 2)}{4}$	N

a. Standard Plane Geometry

For standard plane geometry (see Table 8.1 and Figure 8.1) azimuthal symmetry is assumed in ϕ so that $\underline{\Omega}(\mu, \phi)$ becomes $\underline{\Omega}(\mu)$ and $d\underline{\Omega}$ becomes $2\pi d\mu$. The angular interval $\mu \in [-1, 1]$ is discretized into MM quadrature points μ_m and associated weights w_m ordered as shown in Figure 8.4. Note that the weights, w_m , correspond to $d\mu_m/2$ for this geometry. The angular flux moments, given by Eq. (17a), are approximated by

$$\phi_L(x) \cong \sum_{m=1}^{MM} w_m P_L(\mu_m) \psi_m(x), \quad (21)$$

The discrete-ordinates approximation to the transport equation, i.e., Eq. (3), becomes

$$\mu_m \frac{\partial \psi_m(x)}{\partial x} + \sigma(x) \psi_m(x) = S_m(x) \quad (22)$$

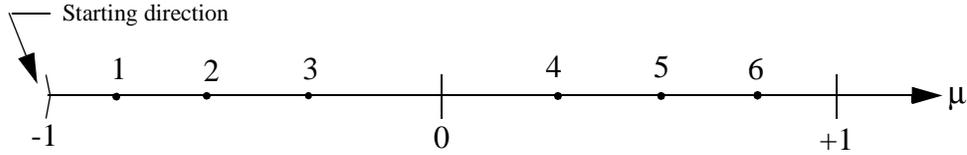


Figure 8.4 Ordering of S_6 directions in plane and spherical geometries

Note: The starting direction only applies to spherical geometry

b. Two-Angle Plane Geometry

For two-angle plane geometry the entire unit sphere of directions is discretized into MM quadrature points (μ_m, ϕ_m) and associated weights ordered as shown in Figure 8.5. The weights, w_m , correspond to $d\Omega_m/4\pi$ for this option. The angular flux moments, given by Eqs. (17a)-(17c), are approximated by

$$\phi_L(x) \cong \sum_{m=1}^{MM} w_m P_L(\mu_m) \psi_m(x), \quad (23a)$$

$$\phi_{C,L}^K(x) \cong \sum_{m=1}^{MM} w_m \psi_m(x) P_L^K(\mu_m) \cos K\phi_m, \quad (23b)$$

$$\phi_{S,L}^K(x) = \sum_{m=1}^{MM} w_m \psi_m(x) P_L^K(\mu_m) \sin K\phi_m \quad (23c)$$

The discrete-ordinates approximation to the transport equation is the same as for standard plane geometry, i.e., Eq. (22) on page 8-29.

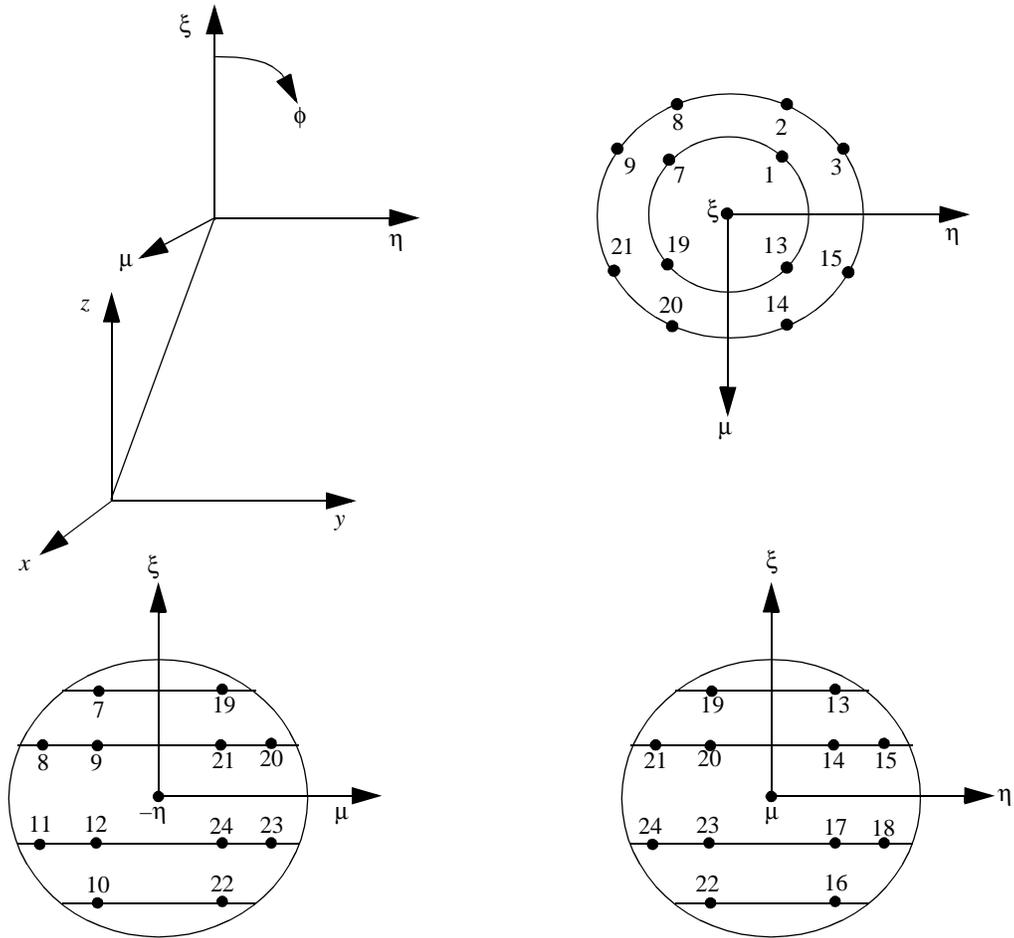


Figure 8.5 Ordering of S_4 directions in two-angle plane geometry*

c. Cylindrical Geometry

For cylindrical geometry (see Table 8.1 and Figure 8.2), the multigroup transport equation, Eq. (3), may be written

$$\mu \frac{\partial(r\psi)}{\partial r} - \frac{\partial(\eta\psi)}{\partial \phi} + r\sigma\psi = rS(r, \underline{\Omega}), \tag{24}$$

where $\psi = \psi(r, \underline{\Omega})$.

*The ordinates in the octant $\mu, \xi < 0, \eta > 0$ are not shown.

For the discrete-ordinates approximation in cylindrical geometry, only one quadrant of the unit sphere is discretized into a set of MM quadrature points (μ_m, η_m) and associated quadrature weights w_m . The ordering of these quadrature points is illustrated in Figure 8.6 for an S_6 quadrature.

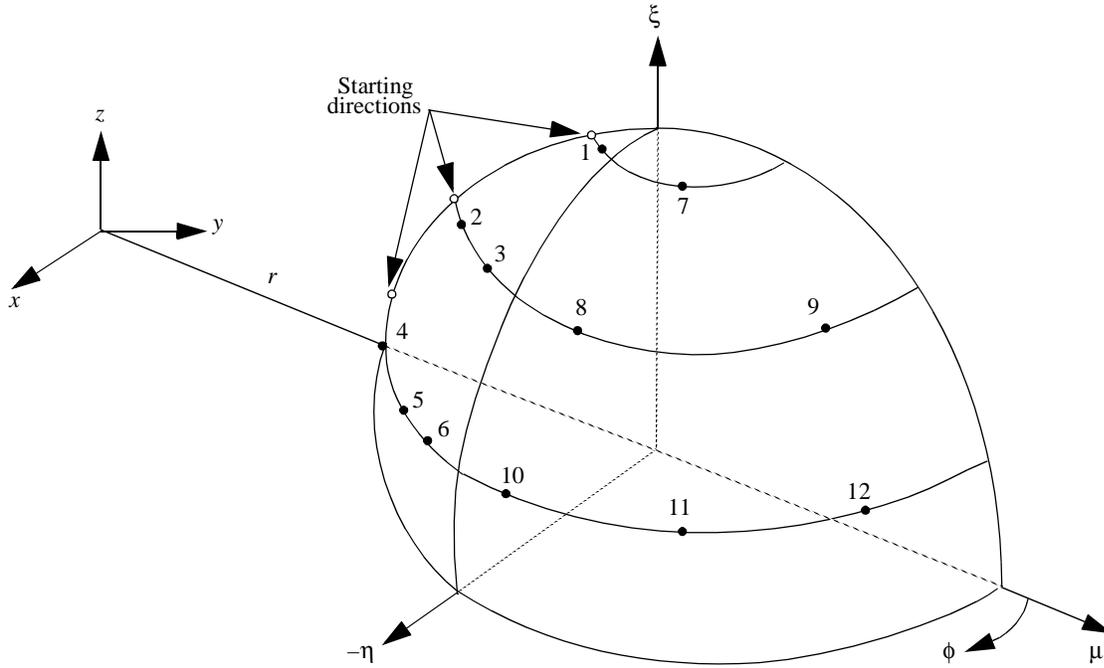


Figure 8.6 Ordering of S_6 directions in cylindrical geometry

As before, $\psi_m(r) \equiv \psi(r, \mu_m, \eta_m)$ represents the average angular flux in $d\Omega_m$ about $\underline{\Omega}_m$ and the angular flux moments for direction m are given by Eqs. (23a)-(23b). In addition, it is necessary to define angular-cell-edge fluxes on a given ξ -level as $\psi_{m-1/2}(r)$ and $\psi_{m+1/2}(r)$. The discrete-ordinates approximation to Eq. (24) can then be written:

$$\begin{aligned} \mu_m \frac{\partial(r\psi_m)}{\partial r} + \left(\frac{\alpha_{m+1/2}}{w_m}\right)\psi_{m+1/2}(r) - \left(\frac{\alpha_{m-1/2}}{w_m}\right)\psi_{m-1/2}(r) + r\sigma\psi_m(r) \\ = rS_m(r) \end{aligned} \quad (25)$$

where the $\alpha_{m-1/2}$ and $\alpha_{m+1/2}$ are angular coupling coefficients. These coefficients satisfy the recursion relation

$$\alpha_{m+1/2} - \alpha_{m-1/2} = -w_m \mu_m, \quad (26)$$

with the requirement that the first ($\alpha_{1/2}$) and last ($\alpha_{M+1/2}$) coefficients on each ξ -level must vanish. It can be shown that Eq. (25) becomes identical to Eq. (24) in the limit of vanishingly small angular intervals. In the output of PARTISN pertaining to the angular quadrature, the quantities $\left(\frac{\alpha_{m+1/2}}{w_m}\right)$ and $\left(\frac{\sigma_{m-1/2}}{w_m}\right)$ are printed out under the headings BETA PLUS and BETA MINUS, respectively.

d. Spherical Geometry

From Table 8.1 the multigroup transport equation, Eq. (3), can be written

$$\mu \frac{\partial(r^2 \psi)}{\partial r} + r \frac{\partial[(1 - \mu^2) \psi]}{\partial \mu} + r^2 \sigma \psi = r^2 S(r, \mu), \quad (27)$$

where azimuthal symmetry in ϕ (see Figure 8.3) has been assumed. The angular domain $\mu \in [-1, 1]$ is discretized into MM quadrature points μ_m and associated weights w_m . Note that in spherical geometry, like standard plane geometry, the w_m correspond to $d\mu_m/2$. The ordering of the quadrature points is illustrated in Figure 8.4. As before, $\psi_m(r) \equiv \psi(r, \mu_m)$ represents the average angular flux in $d\Omega_m (= d\mu_m)$ about $\underline{\Omega}_m$ and the angular flux moments, given by Eq. (17a), are approximated by Eq. (16). In addition, it is necessary to define angular-cell-edge fluxes $\psi_{m-1/2}(r)$ and $\psi_{m+1/2}(r)$. The discrete-ordinates approximation to Eq. (27) is then written as

$$\begin{aligned} \mu_m \frac{\partial(r^2 \psi_m)}{\partial r} + \left[\left(\frac{\beta_{m+1/2}}{w_m} \right) \psi_{m+1/2}(r) - \left(\frac{\beta_{m-1/2}}{w_m} \right) \psi_{m-1/2}(r) \right] r \\ + r^2 \sigma \psi_m(r) = r^2 S_m(r), \end{aligned} \quad (28)$$

where the angular coupling coefficients β must satisfy the recursion relation

$$\beta_{m+1/2} - \beta_{m-1/2} = -2w_m \mu_m, \quad m = 1, \dots, MM, \quad (29)$$

with the requirement from particle conservation that the first ($\beta_{1/2}$) and last ($\beta_{MM+1/2}$) coefficients must vanish. It can be shown¹ that Eq. (28) becomes identical to Eq. (27) in the limit of vanishingly small angular intervals. In the output of PARTISN pertaining to the angular quadrature, the quantities $\left(\frac{\beta_{m+1/2}}{2w_m}\right)$ and $\left(\frac{\beta_{m-1/2}}{2w_m}\right)$ are printed out under the headings BETA PLUS and BETA MINUS, respectively.

e. Starting Directions

For the curved geometries discrete-ordinates Eqs. (25) and (28), there are three variables to be determined at each space position, r : the angular-cell-edge fluxes $\psi_{m-1/2}(r)$ and $\psi_{m+1/2}(r)$ and the average angular flux $\psi_m(r)$. The $\psi_{m-1/2}(r)$ flux can be assumed known (except for $\psi_{1/2}(r)$) from the previous angular mesh-cell computation and assuming continuity at the angular mesh-cell boundaries. The standard diamond-difference² assumption in angle is made to relate the $\psi_{m+1/2}$ to ψ_m , namely,

$$\psi_m(r) = \frac{1}{2}[\psi_{m-1/2}(r) + \psi_{m+1/2}(r)]. \quad (30)$$

Using Eq. (30) to solve for $\psi_{m+1/2}$ and substituting the resulting expression into Eq. (25) or (28), there remains but one equation for the one unknown $\psi_m(r)$.

The assumption that $\psi_{m-1/2}$ is known is correct except for $m = 1$ for which an initial, or starting, condition is required. To achieve this, PARTISN uses special, zero-weighted starting directions in spherical and cylindrical geometries to calculate $\psi_{1/2}(r)$. For spherical geometry this starting direction is the straight-inward direction $\mu = -1$ for which the term $(1 - \mu^2)\psi$ in Eq. (27) vanishes. This yields a special form of Eq. (28) which can be solved for $\psi_{1/2}(r)$. For cylindrical geometry, as shown in Figure 8.6, starting directions corresponding to ordinates directed towards the cylindrical axis, $\eta = 0$, $\varphi = 180^\circ$, are used for each ξ -level to yield special equations for $\psi_{1/2}(r)$ on each ξ -level. At the origin, $r=0$, we also impose an isotropic condition on the angular flux; i.e., $\left.\frac{\partial\psi}{\partial\mu}\right|_{r=0} = 0$. This implies that $\psi_m(0) = \text{constant}$ for all m . The constant is

determined by solving the starting direction equation as $\frac{d\psi}{dr} + \sigma\psi = S$ which applies to spherical geometry and to cylindrical geometry on each ξ level.

Discretization of the Spatial Variable

The spatial domain of the problem is ultimately partitioned into IT fine-mesh intervals of width Δx_i , $i = 1, 2, \dots, IT$ such that $\Delta x_i \equiv x_{i+1/2} - x_{i-1/2}$. Subscripts with half-integer values denote interval boundaries, and integer subscripts denote interval average, or midpoint, values. It is assumed that $x_{i+1/2} > x_i > x_{i-1/2}$. With such a partitioning, space derivatives are approximated by finite differences and, typically, the resulting equations are cast in forms using interval, or mesh, average fluxes, sources, etc.

The spatial discretization of the one-dimensional symmetries is begun by integrating the Eqs. (22), (25), or (27) for slabs, cylinders, and spheres respectively over a spatial mesh cell. The resulting equation is called the discretized balance equation and is the one solved by the PARTISN code. This balance equation for all 3 symmetries is written in the following form:

$$\begin{aligned} & \mu_m (A_{i+1/2} \Psi_{g,m,i+1/2} - A_{i-1/2} \Psi_{g,m,i-1/2}) + \\ & + (A_{i+1/2} - A_{i-1/2}) \frac{(\alpha_{m+1/2} \Psi_{g,m+1/2,i} - \alpha_{m-1/2} \Psi_{g,m-1/2,i})}{w_m} + \\ & + \sigma_{t,g,i} V_i \Psi_{g,m,i} = S_{g,m,i} V_i \end{aligned} \quad (31)$$

$$m = 1, \dots, MM; \quad i = 1, \dots, IT$$

where,

$\Psi_{g,m,i}$ is the spatial cell centered angular flux,

$\Psi_{g,m,i+1/2}$ is the cell edge angular flux at the right cell edge,

$A_{i+1/2}$ is the area of the cell at the right edge, and

V_i is the volume of cell i .

Equation (31) represents IT*MM equations for 3*IT*MM+MM+IT unknowns for each group. The boundary conditions give an additional IT+MM equations. To generate the remainder of the equations, we resort to an approximation which is called diamond differencing. In this case, we specify a relationship between the cell centered and cell edge fluxes in the following way:

$$\Psi_{g,m,i} = 0.5(\Psi_{g,m,i+1/2} + \Psi_{g,m,i-1/2})$$

$$\Psi_{g, m, i} = 0.5(\Psi_{g, m+1/2, i} + \Psi_{g, m-1/2, i}) \quad (32)$$

$$m = 1, \dots, M; \quad i = 1, \dots, IT$$

These equations give the needed $2 \cdot IT \cdot MM$ relationships needed to solve the discretized transport equation. The solution process starts from a known boundary condition which specifies the edge flux at that boundary and follows the particle flow; Eqs. (32) are used to eliminate the unknown edge flux in terms of the known edge and cell centered fluxes. This is then substituted into Eq. (31) to derive an equation for the cell centered flux. Eq. (32) is then used to compute the value for the unknown edge flux from this cell centered flux and the known edge flux. However, for a variety of reasons, the most common of which is that the cells are too large in terms of mean-free-paths, the value for the edge flux can extrapolate to a negative value. This is unphysical given that the source is positive, thus, a fixup is employed. This negative flux fixup sets the unknown edge flux to zero when it extrapolates to a negative value. The balance equation, Eq. (31), is then resolved under this condition in order to maintain particle balance. Thus this scheme is known as diamond differencing with set-to-zero fixup and is the PARTISN basic method for spatial differencing.

TWO-DIMENSIONAL METHODS

The basic discretization methods used in TWODANT are the same as in ONEDANT with the natural extensions to two dimensions. The geometries treated are planar X-Y and polar R- Θ , and cylindrical R-Z, and these are subsets of the symmetries presented in Figs. 12.1 and 12.2 in the section. In the following paragraphs we present some of the details specific to the methods we use in two dimensions.

Some Angular Details in Two Dimensions

The angular variable is treated using discrete ordinates as in one-dimension, but the domain is extended to the hemisphere due to the two-dimensional geometries. The explicit form used in the spherical harmonics expansion of the sources is given in Table 12.5.

Table 8.5 Spherical Harmonics in Two Dimensions

n	Two-Dimensional Geometries P_3
1	$\underline{P_0(\mu)}$
2	$\underline{P_1(\mu)}$
3	$\underline{P_1^1(\mu) \cos \phi}$
4	$\underline{P_2(\mu)}$
5	$\underline{\frac{\sqrt{3}}{3} P_2^1(\mu) \cos \phi}$
6	$\underline{\frac{\sqrt{3}}{6} P_2^2(\mu) \cos 2\phi}$
7	$\underline{P_3(\mu)}$
8	$\underline{\frac{\sqrt{6}}{6} P_3^1(\mu) \cos \phi}$
9	$\underline{\frac{\sqrt{15}}{30} P_3^2(\mu) \cos 2\phi}$

Table 8.5 Spherical Harmonics in Two Dimensions

n	Two-Dimensional Geometries P_3
10	$\frac{\sqrt{10}}{60} P_3^3(\mu) \cos 3\phi$

This table shows the specific case for P_3 expansion which has 10 moments. In general the number of moments is given by $(L+1)(L+2)/2$ where L is the Legendre expansion order.

In PARTISN we have two possible arrangements of the discrete ordinate points on an octant of the unit sphere. One arrangement is triangular in which the first level gets one point, the second gets two, and so forth. The second is a square arrangement in which all levels get the same number of points equal to the number of levels. A fairly complete discussion of the issues involved, especially the concept of levels is in section IV.A of Ref. 4. In Table 12.6 we show the number of quadrature points as a function of S_n order for the two-dimensional symmetries in the triangular and square arrangements.

Table 8.6 Number of Angles Per Octant in Two Dimensions

N	Triangular Arrangement	Square Arrangement
2	1	1
4	3	4
6	6	9
8	10	16
12	21	36
16	36	64
N	$N(N+2)/8$	$N \cdot N/4$

Transport Operator in Two-Dimensional Symmetries

The transport operator for each of the two-dimensional symmetries that PARTISN treats are listed in the following. They are written in conservative form since this is the form that is used to derive the spatially discretized equations solved in the code.

Planar X-Y Symmetry

$$\underline{\Omega}_m \cdot \nabla \psi_m = \mu_m \frac{\partial \psi_m}{\partial x} + \eta_m \frac{\partial \psi_m}{\partial y} \quad (33)$$

where,

$$\mu_m = \hat{e}_x \cdot \underline{\Omega}_m,$$

$$\eta_m = \hat{e}_y \cdot \underline{\Omega}_m.$$

Planar R- θ Symmetry

$$\underline{\Omega}_m \cdot \nabla \psi_m = \frac{\mu_m}{r} \frac{\partial}{\partial r} (r \psi_m) + \frac{\eta_m}{r} \frac{\partial \psi_m}{\partial \theta} - \frac{1}{r} \frac{\partial}{\partial \theta} (\eta \psi_m) \quad (34)$$

where,

$$\mu_m = \hat{e}_r \cdot \underline{\Omega}_m,$$

$$\eta_m = \hat{e}_\theta \cdot \underline{\Omega}_m.$$

Cylindrical R-Z Symmetry

$$\underline{\Omega}_m \cdot \nabla \psi_m = \frac{\mu_m}{r} \frac{\partial}{\partial r} (r \psi_m) - \frac{1}{r} \frac{\partial}{\partial \phi} (\eta_m \psi_m) + \xi_m \frac{\partial \psi_m}{\partial z} \quad (35)$$

where,

$$\mu_m = \hat{e}_r \cdot \underline{\Omega}_m,$$

$$\xi_m = \hat{e}_z \cdot \underline{\Omega}_m.$$

Spatially Discretized Two-Dimensional Transport Equation

The spatially discretized equations for all the symmetries can be written as a balance equation in a single form. The balance equation is derived by integrating the above equations over a spatial mesh cell.

$$\mu_m (A_{i+1/2,j} \psi_{g,m,i+1/2,j} - A_{i-1/2,j} \psi_{g,m,i-1/2,j}) +$$

$$\begin{aligned}
& + \eta_m B_{i,j} (\Psi_{g,m,i,j+1/2} - \Psi_{g,m,i,j-1/2}) + \\
& + (A_{i+1/2,j} - A_{i-1/2,j}) \frac{(\alpha_{m+1/2} \Psi_{g,m+1/2,i,j} - \alpha_{m-1/2} \Psi_{g,m-1/2,i,j})}{w_m} + \\
& + \sigma_{t,g,i,j} V_{i,j} \Psi_{g,m,i,j} = S_{g,m,i,j} V_{i,j}
\end{aligned} \tag{36}$$

where,

$\Psi_{g,m,i+1/2,j}$ is the flux on the right edge of the mesh cell,

$\Psi_{g,m,i,j+1/2}$ is the flux on the top edge of the mesh cell,

$\Psi_{g,m+1/2,i,j}$ is the angular direction edge flux,

$\Psi_{g,m,i,j}$ is the cell center angular flux,

$A_{i+1/2,j}$ is the mesh cell area in the i coordinate direction,

$B_{i,j}$ is the mesh cell area in the j coordinate direction,

$V_{i,j}$ is the mesh cell volume.

We note that in X-Y symmetry, the areas in the i direction, $A_{i+1/2,j}$, are all equal to unity, and hence the term with the angular derivative vanishes.

Equation (36) represents IT*JT*MM equations for 4*IT*JT*MM + (IT+JT)*MM + IT*JT unknowns for each group. The boundary conditions give an additional IT*JT + (IT+JT)*MM equations. As implemented in the PARTISN code, we generate the remainder of the equations by one of two approximations: diamond differencing with set-to-zero fixup or adaptive weighted diamond differencing. In the diamond case, we specify a relationship between the cell centered and cell edge fluxes in the following way:

$$\begin{aligned}
\Psi_{g,m,i,j} &= 0.5(\Psi_{g,m,i+1/2,j} + \Psi_{g,m,i-1/2,j}) \\
\Psi_{g,m,i,j} &= 0.5(\Psi_{g,m,i,j+1/2} + \Psi_{g,m,i,j-1/2}) \\
\Psi_{g,m,i,j} &= 0.5(\Psi_{g,m+1/2,i,j} + \Psi_{g,m-1/2,i,j}) \\
m &= 1,\dots,MM; \quad i = 1,\dots,IT; \quad j=1,\dots,JT
\end{aligned} \tag{37}$$

These equations give the needed 3*IT*JT*MM relationships needed to solve the discretized transport equation. The solution process starts from a known boundary condition which specifies the edge flux at that boundary and follows the particle flow; Eqs. (37) are used to eliminate the unknown edge flux in terms of the known edge and cell centered fluxes. This is then substituted into Eq. (36) to derive an equation for the cell centered flux. Eq. (37) is then used to compute the value for the unknown edge flux from the cell centered and the known edge fluxes. However, as in the one dimensional case, the value for the edge flux can extrapolate to a negative value and so a fixup is employed. This negative flux fixup sets the unknown edge flux to zero if it extrapolates negative, and the balance equation, Eq. (36), is resolved under this condition to maintain particle balance.

The second method that we use for the spatial discretization is adaptive weighted diamond differencing (AWDD). This is based upon a weighted diamond approximation which, rather than fixing up, uses a predictor corrector method to determine the weights that will give a positive edge flux in each of the coordinate directions. The weighted diamond equations are written as:

$$\begin{aligned}
(1 + P_{x,g,m,i,j})\Psi_{g,m,i,j} &= \begin{pmatrix} \Psi_{g,m,i+1/2,j} + P_{x,g,m,i,j}\Psi_{g,m,i-1/2,j} \\ P_{x,g,m,i,j}\Psi_{g,m,i+1/2,j} + \Psi_{g,m,i-1/2,j} \end{pmatrix} \begin{pmatrix} \mu_m > 0 \\ \mu_m < 0 \end{pmatrix} \\
(1 + P_{y,g,m,i,j})\Psi_{g,m,i,j} &= \begin{pmatrix} \Psi_{g,m,i,j+1/2} + P_{y,g,m,i,j}\Psi_{g,m,i,j-1/2} \\ P_{y,g,m,i,j}\Psi_{g,m,i,j+1/2} + \Psi_{g,m,i,j-1/2} \end{pmatrix} \begin{pmatrix} \eta_m > 0 \\ \eta_m < 0 \end{pmatrix} \\
(1 + P_{a,g,m,i,j})\Psi_{g,m,i,j} &= P_{a,g,m,i,j}\Psi_{g,m+1/2,i,j} + \Psi_{g,m-1/2,i,j} \tag{38} \\
m &= 1,\dots,MM; \quad i = 1,\dots,IT; \quad j=1,\dots,JT \\
|P_{x,g,m,i,j}| &\leq 1, \quad |P_{y,g,m,i,j}| \leq 1, \quad |P_{a,g,m,i,j}| \leq 1
\end{aligned}$$



THREE-DIMENSIONAL METHODS

The methods that we use in three dimensions are the same as in two dimensions except with the natural extensions to three dimensions. The S_n quadrature is the same except that eight octants are used as opposed to the four quadrants in 2 dimensions. The spherical harmonics used are the same as the two-angle slab entries from one dimension. The spatial discretization has the same two options of diamond with set-to-zero fixup and the AWDD methods described above in the two-dimensional methods section.

REFERENCES

1. B. G. Carlson and K. D. Lathrop, "Transport Theory-Method of Discrete Ordinates," in Computing Methods in Reactor Physics, H. Greenspan, C. N. Kelber and D. Okrent, Eds. (Gordon and Breach, New York, 1968), Chap. III, p. 185.
2. G. I. Bell and S. Glasstone, "Discrete Ordinates and Discrete S_N Methods," in Nuclear Reactor Theory, (Van Nostrand Reinhold, New York, 1970), Chap. 5, p. 211.
3. R. E. Alcouffe, "Diffusion Synthetic Acceleration Methods for the Diamond-Difference Discrete-Ordinates Equations," *Nucl. Sci. Eng.* **64**, 344 (1977).
4. R. D. O'Dell and R. E. Alcouffe, "Transport Calculations for Nuclear Analysis: Theory and Guidelines for Effective Use of Transport Codes," Los Alamos National Laboratory report LA-10983-MS (September 1987).
5. R. E. Alcouffe, "An Adaptive Weighted Diamond Differencing Method for Three-Dimensional XYZ Geometry," *Trans. Am. Nuc. Soc.* **68**, Part A, 206 (1993).

METHODS CHAPTER INDEX

A

Acceleration	
Diffusion synthetic(DSA).....	8-14
Anisotropy	
Cross section scattering source.....	8-22
Approximation	
Discrete ordinates	8-12
Multigroup.....	8-11

B

Boltzmann transport equation	8-9
------------------------------------	-----

D

Diffusion synthetic acceleration(DSA)	
General method	8-14
Discrete ordinates approximation	8-12
Discrete ordinates equation	
In one dimension	8-27
Discretization of the spatial variable	
In one dimension	8-32
In triangular geometry	8-51
In two dimensions.....	8-37
In TWOHEX	8-51
Divergence operator in one dimension	8-21

E

Expansion of the inhomogeneous source.....	8-26
Expansion of the scattering source	8-22

G

Geometry	
Symmetries treated in ONEDANT.....	8-19

I

Iteration procedure	
General method	8-14

M

Methods	
Chapter on solution methods	8-1
Diffusion synthetic	8-14
Monte Carlo/Discrete Ordinates.....	8-40
Monte Carlo/Discrete Ordinates	
Methods	8-40
Multigroup approximation.....	8-11

O

Operator	
Divergence operator in one dimension.....	8-21
Transport, in two dimensions	8-36

Q

Quadrature points	
Number in one dimensional geometries.....	8-27
Number in two dimensions.....	8-36
Ordering in one dimension	8-28, 8-29, 8-30
Starting directions.....	8-32

S

Solution	
Chapter on solution methods	8-1
Source	
Anisotropy in	8-22
Spatial discretization	
Adaptive weighted diamond method(AWDD).....	8-38
Diamond method	8-38
In one dimension	8-32
In two dimensions.....	8-37
Spherical harmonics	
Expansion of the inhomogeneous source	8-26
Expansion of the scattering source	8-22

In one dimensional geometries 8-25
In two dimensions..... 8-35
Number in one dimensional geometries 8-24

T

Theory, see Methods
Transport equation, Boltzmann..... 8-9
Transport operator in two dimensions 8-36

CODE STRUCTURE

Transport Methods Group, CCS-4
Los Alamos National Laboratory

9

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	9-3
LIST OF TABLES.....	9-5
OPERATION OF THE CODE SYSTEM	9-7
Programming Practices and Standards.....	9-7
Language	9-7
Structure	9-7
Standard Interface Files.....	9-7
Data Management and Transfers.....	9-8
Central Memory Restrictions	9-8
Word Size	9-9
CODE PACKAGE STRUCTURE	9-10
Input Module.....	9-16
Solver Modules	9-17
Edit Module	9-18
PIECEWISE EXECUTION.....	9-19
Module Execution Control.....	9-19
Submodule Execution Control (File Generation Suppression).....	9-20
STACKED RUNS	9-25
REFERENCES	9-27
CODE STRUCTURE CHAPTER INDEX.....	9-28

LIST OF TABLES

Table 19.1: Files Read and Written.....	9-10
Table 19.2: Structure of the Input Module.....	9-13
Table 19.3: Structure of the Solver Module.....	9-14
Table 19.4: Structure of the Edit Module.....	9-15

OPERATION OF THE CODE SYSTEM

The PARTISN code package is a modular computer program package designed to solve the time-independent or dependent, multigroup discrete ordinates form of the Boltzmann transport equation in several different geometries. It was developed as a modular code package consisting of three modules: an Input Module, a Solver Module, and an Edit Module.

In this chapter is provided a discussion of the general programming practices and standards used in the code package, a description of the code structure, and overviews of the three modules comprising the package.

Programming Practices and Standards

In general, the programming standards and practices recommended by the Committee on Computer Code Coordination (CCCC)^{1,2} have been followed throughout the development of the PARTISN package. By following these practices and standards, problems associated with exporting and implementing the code in different computing environments and at different computing installations are minimized. This section provides a brief summary of the CCCC programming practices and standards used in PARTISN.

Language

The programming language is standard Fortran 90 as defined by the ANSI standard X3.198-1992.³ The parallel programming language is MPI v1.1.

Structure

The code is structured in a form that separates the input and the output (or edit) functions from the main calculational (or solver) sections of the code. A more complete description of the code structure is provided in “CODE PACKAGE STRUCTURE” starting on page 9-10.

Standard Interface Files

PARTISN makes use of interface files to transmit data between and within its modules. These interface files are binary, sequential data files. Standard interface files are interface files whose structure and data-content formats have been standardized by the

CCCC. Code-dependent interface files are files whose structure and data-content formats have not been standardized.

The following CCCC standard interface files are accepted, created, or otherwise used in PARTISN: ISOTXS, GRUPXS, GEODST, NDXSRF, ZNATDN, SNCONS, FIXSRC, RTFLUX, ATFLUX, RZFLUX, RAFLUX, and AAFLUX. File descriptions for these files are provided in Ref. 1.

The following code-dependent, binary interface files are used in PARTISN: MACRXS, BXSLIB, FISSRC, RMFLUX, AMFLUX, AZFLUX, RZMFLX, AZMFLX, RAFLXM, AAFLXM, UCFLUX, LNK3DNT, SNXEDT, ADJMAC, SOLINP, EDITIT, ASGMAT. ASCII code-dependent files produced by or usable by the code package are MACBCD, XSLIBB, XSLIBF, XSLIBE, ARBFLUX, EDTOUT, and EDTOGX. File descriptions for these code-dependent files are provided in the chapter “FILE DESCRIPTIONS” starting on page 11-1.

The use of the above interface files is described in “CODE PACKAGE STRUCTURE” starting on page 9-10 of this chapter.

Data Management and Transfers

PARTISN is designed with data-management techniques to accommodate, as efficiently as possible, the transfer of the large amounts of data frequently needed for solving large problems. Data management in the code involves the reading and writing of sequential data files, a flexible capability to block data, and if needed, use of multilevel data-management/transfers using random-access files.

The CCCC standardized subroutines SEEK, REED, and RITE are used for data transfers involving binary, sequential data files. A description of these routines is provided in Ref. 1. These routines have been extended to allow for parallel I/O on some platforms.

For multilevel data transfer using random (direct)-access files, the CCCC procedures have been implemented in the DANTSYS package. The standardized subroutines DOPC, CRED/CRIT, DRED/DRIT are used to effect multilevel data transfers using random-access files. A description of these procedures and subroutines is provided in Ref. 1.

Central Memory Restrictions

PARTISN was originally designed to be operable within a 50,000-word central memory. But now memory is obtained from the heap, and thus is limited only by the size of the heap.

Word Size

The code is designed to be easily converted from its basic long-word computer form to a form for use on short-word computers. (On a long-word computer, a six-character Hollerith word is a single-precision word, while on a short-word computer, it is a double-precision word.)

CODE PACKAGE STRUCTURE

PARTISN consists of three major, functionally independent modules: an Input Module, a Solver Module, and an Edit Module. These modules are linked by means of binary interface files. The Input Module processes any and all input specifications and data and, if required, generates the binary files for use by the Solver and/or Edit modules. The Solver Module performs the transport calculation and generates flux files for use by the Edit Module.

The Solver Module also generate other interface files for use by other codes or for subsequent calculations by the Solver Module. The Edit Module performs cross-section and response function edits using the flux files from the Solver Module.

A complete list of the interface files accepted, used, and generated by the modules is shown in Table 9.1 The table indicates which modules read or write a particular file. In the table, the notation A means always, O means optionally. Thus for the Edit Module, we see an A for the GEODST file indicating that if the Edit Module is invoked, it will always need a GEODST file to read. An entry of A for output is subject to the provision that the information to be put on the file is available. If there is geometry input, then the Input Module will always write a GEODST file. But, if there was no geometry input, the Input Module would not write the GEODST file.

Table 9.1 Files Read and Written

Information Type	File	Input Module		Solver Module		Edit Module	
		Read	Write	Read	Write	Read	Write
Geometry Information	GEODST	O	A	A	O	A	
	Card Images	O					

Table 9.1 Files Read and Written (Cont.)

Information Type	File	Input Module		Solver Module		Edit Module	
		Read	Write	Read	Write	Read	Write
Cross Sections	ISOTXS	O					
	GRUPXS	O					
	MENDF	O					
	MACRXS / ADJMAC	O	A	A			
	MACBCD	O	O				
	XSLIB	O					
	XSLIBE		O				
	XSLIBF		O				
	BXSLIB	O	O				
	XSLIBB	O	O				
	SNXEDT		A			O	
	Card Image	O					
Material Mixing	NDXSRF/ ZNATDN	O	A	A		O	
	Card Images	O					
	LNK3DNT			O		O	
Assignment of Materials to Zones	ASGMAT		A	A	O	O	
	Card Images	O					
Solver Module Input	SOLINP	O	A	A			
	Card Images	O					
Quadrature	SNCONS			O	A		
Inhomoge- neous Sources	FIXSRC			O	O		
	Card Images	O					
Edit Module Input	EDITIT		A			A	
	Card Images	O					

Table 9.1 Files Read and Written (Cont.)

Information Type	File	Input Module		Solver Module		Edit Module	
		Read	Write	Read	Write	Read	Write
Other Output Files	RTFLUX/ ATFLUX			O	A	A	O
	RAFLUX/ AAFLUX				O		
	RZFLUX/ AZFLUX						O
	RMFLUX/ AMFLUX			O	O	O	
	RAFLXM/ AAFLXM				O		
	RZMFLX/ AZMFLX						O ^a
	FISSRC				O		
	EDTOUT						O
	EDTOGX						O
UCFLUX			O	O			

a. Requires an RMFLUX or AMFLUX file from the Solver.

A segmented structure is used in PARTISN for implementing the modules. Such a structure involves the use of a main driver together with input, solver, and edit segments.

The main program, DRIVER, controls the calling of the primary segments, together with those service subroutines used by more than one segment.

The first segment constitutes the Input Module. It is structured into a driver routine, INPT10, plus twelve secondary sections as shown in Table 9.2. Each of the secondary sections performs a unique function so that the Input Module itself is constructed in a modular form.

Table 9.2 Structure of the Input Module

Routine	Function
INPT10	Input Module Driver: controls the flow of the code by calling one or more of the secondary somebodies below.
INPT11	Controls code setup and storage allocation.
INPT12	Controls geometry data processing
INPT13	Controls cross-section library processing for XSLIB, MENDF, XSLIBB, and MACBCD library forms.
INPT14	Controls mixing specification processing.
INPT15	Controls GRUPXS cross-section library processing.
INPT16	Controls ISOTXS cross-section library processing.
INPT17	Controls BXSLIB cross-section library processing.
INPT18	Controls Solver Module input data processing.
INPT19	Controls Edit Module input data processing.
INP110	Controls cross-section balancing operation.
INP111	Controls adjoint reversals.
INP112	Controls GEODST file post processing.

The second segment constitutes the Solver, or calculational, module. It consists of a driver routine plus additional secondary sections. The structure of the PARTISN Solver Module is depicted in Table 9.3.

Table 9.3 Structure of the Solver Module

Routine	Function
TIGFA03D	Solver Module Driver; controls the flow of the code by calling one or more of the secondary submodules below.
TINP213D	Controls module initializations.
TINP223D	Controls flux guess and inhomogeneous source processing.
TINP233D	Controls quadrature selection.
TINP243D	Checks spatial mesh input for consistency.
TGND253D	Calculates initially required functions and grid structures; calculates the first collision source if required; initializes the fission source.
TRANSO3D	Controls the inner iteration.
DIFFO3D	Controls the outer iteration.
TOT283D	Controls final Solver Module printing.
TOT293D	Controls binary file preparation.

The third segment is the Edit Module. It currently consists of a driver routine, OUTT30, plus four secondary sections as shown in Table 9.4.

Table 9.4 Structure of the Edit Module

Routine	Function
OUTT30	Edit module driver; controls the flow of the code by calling one or more of the secondary submodules below.
OUTT31	Controls reaction rate calculations.
OUTT32	Controls power normalization, edit zone averaging, and output file preparation.
OUTT33	Controls a spatial mesh collapse determination.
OUTT34	Controls the mass edit request on the coarse mesh or edit zones.

A fourth segment is used in PARTISN. This fourth segment provides highlights of the just-executed run as an aid to the user. These highlights are a printed summary of some of the pertinent facts, options, and decisions encountered during the run along with storage and run time information. This segment is not considered to be a module in the sense of the first three segments.

Input Module

The Input Module performs the necessary activities for processing all input data required for the execution of the Solver and/or Edit Modules. These activities include the reading of input data and the creation of binary interface files. The latter activity may require a certain degree of data processing. Each of these activities is discussed below.

In performing the reading-of-input-data activity, the Input Module accepts standard interface files (binary), code-dependent binary interface files, or card-images for its input. These are listed in Table 9.1. As is indicated in the table, input data to the code can be provided in several different forms and many combinations of forms to provide a great deal of flexibility to the user. These input data are described in the appropriate User's Guide for the code of interest.

The second major activity in the Input Module is the creation of binary interface files containing all input data. These files are subsequently used as the sole means of transmitting data to either the Solver or Edit Modules. The files emerging from the Input Module are given in Table 9.1 and take the form of either CCCC standard interface files or code-dependent interface files. In this file-creation activity, the Input Module is called on to perform several types of tasks. As an example, the only form in which geometry-related information emerges from the Input Module is in the form of a GEODST standard interface binary file. If a user supplies geometry-related input by means of card-image input, the Input Module reads this input, translates the data into a GEODST-compatible form, and creates the resulting GEODST file. On the other hand, if the geometry-related information is supplied by the user through an already existing GEODST or LNK3DNT file, the Input Module is required to do nothing.

A second, more complex, example of the function of the Input Module involves the mixing of isotopes, or nuclides, to create materials which are subsequently assigned to physical regions in the problem (called zones) to define the macroscopic cross-section data for the zones. For this example, it will be assumed that the user selects card-image input as the form for the Input Module. First, the isotope mixing specifications appropriate for the desired materials are input via card-image. The Input Module reads this data, translates the data and creates the two standard interface files NDXSRF and ZNATDN as shown in Table 9.1. These two files appear as output from the Input Module. Assuming next that the isotope cross sections are provided by the user as a card-image library, the Input Module reads this library (in isotope-ordered form) and also reads the just-created NDXSRF and ZNATDN files. The mixing specifications provided by the latter files are applied to the isotopic cross-section data to generate material cross sections which are written, in group order, to a code-dependent binary file named MACRXS. (A group-ordered file named SNXEDT for use by the Edit Module is also created at this time but will not be considered in this example.) The MACRXS file becomes the sole source of

cross-section data to the Solver Module if the Solver calculation is to be a forward, or regular, calculation. If an adjoint calculation is to be performed by the Solver, the Input Module re-reads the MACRXS file, performs the adjoint reversals on the cross sections, and creates the code-dependent binary file named ADJMAC containing the adjoint-reversed material cross sections for use by Solver.

Solver Module

The Solver Module of PARTISN has the function of effecting numerical solutions of the multigroup form of the neutral-particle steady-state Boltzmann transport equation. The discrete-ordinates approximation is used for treating the angular variation of the particle distribution and the diamond-difference scheme⁴ or the adaptive weighted diamond difference scheme in 2 and 3-d,⁵ is used for phase space discretization.

In solving the transport equation numerically, an iterative procedure is used. This procedure involves two levels of iteration referred to as inner and outer iterations. The acceleration of these iterations is of crucial importance to transport codes in order to reduce the computation time involved. The PARTISN Solver Module employs the diffusion synthetic acceleration method developed by Alcouffe,⁶ an extremely effective method for accelerating the convergence of the iterations. A relatively detailed development of the solution methods used in the Solver Module is provided in the chapter “PARTISN — METHODS MANUAL” starting on page 8-1.

The Solver Module is essentially a free-standing entity, and input to and output from the module is in the form of binary files together with limited printed output. The binary interface files used as input to the Solver Module are listed in Table 9.1. The files required for execution of the module are a GEODST or LNK3DNT interface file together with the code-dependent interface files MACRXS or ADJMAC, ASGMAT, and SOLINP. Optional files, which may be input to the Solver Module, are the standard interface files SNCONS, RTFLUX or ATFLUX, RMFLUX or AMFLUX, and FIXSRC.

The output from the Solver Module always consists of the scalar flux standard interface file RTFLUX (or ATFLUX if an adjoint problem were run), the standard interface file SNCONS, and user-selected printed output. If desired by the user, the angular flux standard interface file RAFLUX (or AAFLUX, if an adjoint problem were run) will be produced. If an inhomogeneous source problem were run, a FIXSRC standard interface file would be produced. If desired by the user, the angular flux moments code-dependent interface file RMFLUX (or AMFLUX, if an adjoint problem were run) would be produced.

Edit Module

The function of the Edit Module is to produce the printed edit-output selected by the user. Edit-output refers to information which is obtained from data contained on one or more interface files but which generally requires manipulating or processing of the data. An example of the edit-output is a microscopic reaction-rate distribution, $\sigma\phi$, where σ is a particular multigroup, microscopic cross section for a particular isotope or nuclide and ϕ is the multigroup scalar flux distribution obtained from the Solver Module. In this example, data from both a cross-section interface file and a scalar flux file are required to be recovered and multiplied, and the product printed.

The Edit Module is an essentially free-standing module accepting only interface files as input and producing printed output. The required input files for execution of the Edit Module are the code-dependent binary interface file EDITIT and the standard interface files RTFLUX (or ATFLUX) and GEODST as shown in Table 9.1. Optional input files are the standard interface files NDXSFR and ZNATDN and the code-dependent files SNXEDT and ASGMAT. The code-dependent files are produced by the Input Module.

PIECEWISE EXECUTION

As previously described, PARTISN is comprised of three major functionally independent modules: the Input Module, the Solver Module, and the Edit Module. The modules are linked solely by means of binary interface files. The Input Module processes any and all card-image input and, if required, generates the binary interface files for use by the Solver and/or Edit Modules. The Input Module itself is constructed in a modular form and thus is comprised of submodules, each of which performs a unique function related to the generation of certain binary interface files. The Solver Module accepts the appropriate interface files produced by the Input Module (or any other computer code capable of producing such interface files), performs the transport calculation, and generates standard interface flux files for use by the Edit Module (or other computer codes). The Edit Module accepts the appropriate standard and code-dependent interface files and performs cross-section and user-input response function edits.

With the modular construction of the code package and the interface file linkage between modules and submodules, there is a great deal of flexibility provided in the execution flow of a particular computer run. For example, the processing of the input, the execution of the transport solution, and the editing of the results of the solution can be effected as three separate and distinct computer runs and not as a single (perhaps expensive) run. All that need be done is to save the appropriate interface files from each partial execution run and to make these files available to the module to be executed in the next partial execution. This mode of operation enables the user, for example, to process his problem input specification (mixing of nuclides, cross-section preparation, geometry specification, etc.) and to analyze his input before committing it to the Solver Module. If errors are discovered in, say, the geometry specification, the user can correct the errors in the card-image input and simply rerun the geometry-related submodule of the Input Module. When certain that the input is correct, the user can then execute the Solver Module. Following the successful running of the Solver Module, one or more executions of the Edit Module can then be independently made.

In this chapter are provided details for controlling the execution of selected modules and submodules in the code package.

Module Execution Control

The execution of each of the three major modules in the code package (Input, Solver, and Edit Modules) can be independently controlled as described below.

1. Input Module Execution Control.

The Input Module may be thought of as an interface file generating module. It processes card-image input and creates binary interface files as shown in a previous table. Accordingly, if any Block-II through Block-VI card-image input is provided, the Input Module will be executed and the appropriate interface files created.

The execution of the Input Module will be suppressed if there is no card-image input provided other than Block-I input. If the Input Module is not executed, none of its interface files will be created in that execution of the code.

2. Solver Module Execution Control.

Execution of the Solver Module will be attempted if both the following conditions are met: (i) a SOLINP binary interface file exists and is available to the Solver Module, and (ii) the Block-I input parameter NOSOLV is zero.

The Solver Module will not be executed if the Block-I input parameter NOSOLV is set to unity.

Alternatively, since the Input Module creates the SOLINP interface file solely from card-image input provided in Block-V of the input, the user can suppress the execution of the Solver Module by simply omitting all Block-V data from the card-image input and ensuring that there is no other SOLINP file present.

3. Edit Module Execution Control. Execution of the Edit Module will be attempted if both of the following conditions are met: (i) an EDITIT binary interface file exists and is available to the Edit Module and (ii) the Block-I input parameter NOEDIT is zero.

The Edit Module will not be executed if the Block-I input parameter NOEDIT is set to unity.

Alternatively, since the Input Module creates the EDITIT interface file solely from card-image input provided in Block-VI of the input, the user can suppress the execution of the Edit Module by simply omitting all Block-VI data from the card-image input and ensuring that there is no other EDITIT file present.

Submodule Execution Control (File Generation Suppression)

The Input Module is constructed in submodular form. Each submodule has a unique interface file-creation function, and each has its associated card-image input. Also associated with each submodule is a Block-I input flag to turn off, or suppress, the execution of that submodule. The control of the execution of the Input Module submodules is described below.

1. Geometry Submodule Execution Control.

The geometry submodule creates a GEODST standard interface file¹ from the Block-II card-image input data described in any of the User's Guides. This submodule will be executed and a GEODST file created by (i) setting (or defaulting) the Block-I input parameters NOGEOD to zero and (ii) providing Block-II input data in the card-image input "deck" or file.

The geometry submodule will not be executed (no GEODST file will be created) if (i) the Block-I input parameter NOGEOD is set to unity or (ii) all Block-II input is omitted from the card-image input "deck."

2. Mixing Submodule Execution Control.

The mixing submodule creates the standard interface files NDXSRF and ZNATDN¹ from the Block-IV card-image input data found in the MATLS array and, optionally, the PREMIX array as described in a User's Guide.

The mixing submodule will be executed and the NDXSRF and ZNATDN files created by both (i) setting (or defaulting) the Block-I input parameter NOMIX to zero and (ii) providing card-image input through the MATLS array in Block-IV.

The mixing submodule will not be executed if (i) NOMIX is set to unity or (ii) the MATLS input array is omitted from the Block-IV card-image input or (iii) LIB= MACRXS or LIB= MACBCD in Block-III.

3. Assignment-of-Materials-to-Zones Submodule Execution Control.

The assignment-of-materials-to-zones submodule creates the code-dependent interface file ASGMAT from the Block-IV card-image data found in the ASSIGN array. Details on the assignment of materials to zones are given in the User's Guides.

This submodule will be executed and the ASGMAT file created by both (i) setting (or defaulting) the Block-I input parameter NOASG to zero and (ii) providing card-image input through the ASSIGN array in Block-IV.

The submodule will not be executed (no ASGMAT file created) if either (i) the Block-I input parameter NOASG is set to unity or (ii) the ASSIGN input array is omitted from the Block-IV card-image input.

4. Working-Cross-Section-File Submodule Execution Control.

The working-cross-section-file submodule creates the code-dependent interface files MACRXS and SNXEDT.

The working-cross-section-file submodule will be executed and the MACRXS and SNXEDT files created if both the following conditions are met: (i) the Block-I input parameter NOMACR is set (or defaulted) to zero, and (ii) the

Block-III input parameter LIB is not specified as LIB= MACRXS or LIB= MACBCD.

The submodule will not be executed (no MACRXS and SNXEDT files created) if either (i) the Block-I input parameter NOMACR is set to unity or (ii) the Block-III input parameter LIB is specified as LIB= MACRXS or LIB= MACBCD.

Since the formation of the working cross-section files MACRXS and SNXEDT can be quite time-consuming for large multigroup cross-section libraries, it is frequently advantageous to save the computationally ordered MACRXS and SNXEDT files created in one run for use in subsequent runs. Through the use of the NOMACR parameter in Block-I or the LIB= MACRXS parameter in Block-III of the input, the user can easily suppress the re-execution of the working-cross-section-file submodule in subsequent code executions.

5. SOLVER-Input-File Submodule Execution Control.

The Solver-input-file submodule processes the Block-V card-image input and creates the code-dependent interface file SOLINP for use by the Solver Module.

This submodule will be executed and the SOLINP file created if both (i) the Block-I input parameter NOSLNP is set (or defaulted) to zero and (ii) Block-V card-image input is supplied.

The Solver-input-file submodule will not be executed (no SOLINP file created) if either (i) the Block-I input parameter NOSLNP is set to unity or (ii) all Block-V card-image input is omitted from the input “deck.”

6. Edit-input-File Submodule Execution Control.

The EDIT-input-file submodule processes the Block-VI card-image input and creates the code-dependent interface file EDITIT for use by the EDIT module of DANTSYS.

The EDIT-input-file submodule will be executed and the EDITIT file created if both (i) the Block-I input parameter NOEDTT is set (or defaulted) to zero and (ii) Block-VI card-image input is supplied.

This submodule will not be executed (no EDITIT file created) if either (i) the Block-I input parameter NOEDTT is set to unity or (ii) all Block-VI card-image input is omitted from the input “deck.”

7. Adjoint-Reversal Submodule Execution Control.

The adjoint-reversal submodule processes the MACRXS code-dependent cross-section interface file and creates the code-dependent interface file ADJMAC, the adjoint-reversed counterpart to the MACRXS file. This is described in “Adjoint Computations” on page 3-42.

The adjoint-reversal submodule will be executed if both (i) the Block-I input parameter NOADJM is set (or defaulted) to zero and (ii) the Block-V input quantity ITH is set to unity.

The submodule will not be executed (no ADJMAC file created) if either (i) the Block-I input parameter NOADJM is set to unity or (ii) the Block-V input quantity ITH is set to zero.



STACKED RUNS

It is possible to run more than one problem in a single execution of the code by stacking the problem-specification input “decks.” In the context of this discussion the term “deck” refers to all card-image input necessary for a problem. To run more than one problem, the input file is created with two or more problem decks separated from one another by a single card-image record containing the entry

```
]eof
```

beginning in column 1. Thus, a single input file containing the specification decks for three separate problems would be constructed as follows:

```
Problem 1 "deck"  
]eof  
Problem 2 "deck"  
]eof  
Problem 3 "deck"
```

The code output for each of these problems will appear consecutively in a single output file.

Caution: Nonunique interface files created in one problem (for example, an RTFLUX scalar flux file) will be overwritten and lost when the next problem in the stack is executed.

Should a fatal error occur in a problem, the input for any remaining problems will be ignored.

REFERENCES

1. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
2. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physics Codes, Version III," Los Alamos Scientific Laboratory report LA-5486-MS (February 1974).
3. American National Standard Programming Language Fortran, ANSI X3.198-1992, American National Standards Institute, Inc., New York, NY 10018.
4. G. I. Bell and S. Glasstone, "Discrete Ordinates and Discrete S_n Methods," in **Nuclear Reactor Theory**, (Van Nostrand Reinhold, New York, 1970), Chap. 5, pp. 232-235.
5. R. E. Alcouffe, "An Adaptive Weighted Diamond-Differencing Method for Three-Dimensional XYZ Geometry." *Trans. Am. Nuc. Soc.* **68**, Part A (1993).
6. R. E. Alcouffe, "Diffusion Synthetic Acceleration Methods for the Diamond-Difference Discrete-Ordinates Equations," *Nucl. Sci. Eng.* **64**, 344 (1977).

CODE STRUCTURE CHAPTER INDEX

E

Edit Module	
Function of.....	9-17
Execution of code	
Multiple runs	9-25
Piecewise	9-19

F

Files	
Standard interface.....	9-7
Suppressing writing	9-20
Where read, written	9-9

I

Input Module	
Function of.....	9-16

M

Multiple runs	9-25
---------------------	------

P

Piecewise execution of code	9-19
-----------------------------------	------

S

Solver Module	
Function of.....	9-17
Stacked runs	9-25
Structure	
Of code	9-9

ERROR MESSAGES

Transport Methods Group, CCS-4
Los Alamos National Laboratory

10

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	10-3
INPUT ERRORS	10-5
Examples of Errors and Resulting Messages	10-5
Sample Error 1. Misspelled Input Array Name.....	10-5
Sample Error 2. Input Block Terminator Omitted.....	10-6
Sample Error 3. Invalid Entry for Block-I Input Parameter.....	10-7
Sample Error 4. Incorrect Number of Entries in an Input Array.....	10-8
Sample Error 5. Misplaced Array Identifier.....	10-8
COMMENTS REGARDING MULTIPLE ERRORS	10-11
IMPLEMENTATION ERRORS	10-13
WARNINGS.....	10-15
ERRORS CHAPTER INDEX	10-16

INPUT ERRORS

A comprehensive error-checking capability has been provided in the PARTISN code package. Most of the checks are in the Input Module to ensure that the input data are correct, insofar as the code can determine, before execution of the problem commences. Other checks are made in the Solver and Edit Modules to ensure that the modules are executing the desired problem properly.

Input errors will cause the run to terminate before entering the Solver Module. But one important feature of the error diagnostics in the Input Module is that an error will normally not cause an immediate termination. Instead, the code will attempt to process the remaining data in the offending input Block and/or in remaining input Blocks. Only after all remaining input has been processed (if possible) will the run will be terminated.

Error messages are normally provided in at least two places in the output. The first error message is printed at the time that the error was detected by the code. Such messages will be imbedded in the printed output, but they are clearly marked for easy spotting. The second error message will normally occur in the RUN HIGHLIGHTS provided at the end of the printed output. These RUN HIGHLIGHTS provide a printed summary of the code package execution. The user is encouraged to always check the RUN HIGHLIGHTS following a run to quickly ascertain if the completed run did what it was supposed to.

Examples of Errors and Resulting Messages

Several examples of common input errors and the resulting error message printouts are provided below.

Sample Error 1. Misspelled Input Array Name

A common input error is that of misspelling the name of an input array. In this example, the Block-II card-image input array XMESH has been misspelled as XMESSH. The Input Module is thus presented with an unrecognizable and undefined array name resulting in the following error message:

```
*error* card          7 *2 4 6 8(1)2 4 6 8(2)2 4 6 8(3)2 4 6 8(4)2
                      xmessh=0.0 5.0 xints=5 zones=1 t

*error* array name xmessh array number -1
```

```
*error* columns 2 - 8
```

```
undefined array name
```

The first line of the error message indicates that an error was found on input card 7. This is followed by the card-image column numbers. Directly below this, the card-image is reproduced. The third line indicates that the array name XMESSEH is in error and that this array has been given a number -1. (Acceptable arrays are given positive integer identification numbers by the code.) The next line says that the error occurred in columns 2 through 8 on the card-image. Finally, the message that the array name is undefined is provided.

This error is severe enough that further processing of the input is not attempted and no RUN HIGHLIGHTS are produced.

Sample Error 2. Input Block Terminator Omitted

As described in the chapter “FREE FIELD INPUT REFERENCE” starting on page 5-1, each card-image input Block must be terminated with a delimited T, the Block terminator. In this example this terminal “T” has been omitted from the end of the Block-I card-image input. The offending portion actual card-image input for this case is shown below.

```
*****
*
* ...listing of cards in the input stream...
*
* 1. 2 0
* 2. Sample ONEDANT input to display error messages
* 3. Error 2 Terminal "t" omitted from Block I
* 4. /***** B L O C K I *****/
* 5. igeom=slab ngroup=30 isn=8 niso=16 mt=1 nzone=1 im=1 it=5
* 6. /***** B L O C K II *****/
* 7. xmesh=0.0 5.0 xints=5 zones=1 t
* 8. /***** B L O C K III *****/
* 9. lib= bxslib t
* 10. /***** B L O C K IV *****/
* 11. matls=matt 92235.50 assign=matls t
*
*****
```

Note that there is no delimited T at the end of line 5. As a result of this omission the following message is printed:

```
*****
**error** current block contains arrays belonging to other blocks
*****
```

```

*
* no. of    from
* arrays   block
*
*      8      i
*      3      ii
*      0      iii
*      0      iv
*      0      v
*      0      vi

```

```

*****
***error** in block identification
*****

```

The first line of the message indicates that the Input Module has finished reading the card-image input that it thinks belongs in Block-I. (The code has actually read the 8 array entries on line 5 of the input but has continued reading until it found the terminal T following the three array entries belonging to Block-II on line 7 of the input.) The second line of the error message indicates that arrays that do not belong in Block-I have been found. Next is printed a table indicating that eight arrays from Block-I and three from Block-II were discovered in the Block-I card-image reading process. The final error message of an error in Block identification is self-explanatory.

It should be noted that when arrays from other Blocks are found in any given Block, the code will terminate execution immediately and no RUN HIGHLIGHTS are provided.

Sample Error 3. Invalid Entry for Block-I Input Parameter.

As described in any of the user's guides under "Block-I Details: Dimensions and Controls," certain card-image input is always required in Block-I for a PARTISN code execution. Specifically, the eight parameters IGEOM, NGROUP, ISN, NISO, MT, NZONE, IM, and IT are required to be entered as positive integers. In this example, one of these parameters, IM, has been incorrectly entered with a value of zero. It should be noted that if one of these parameters is omitted altogether, the code will default its value to zero.

The code prints the following fatal error message:

```

*****
***error** block i entry .le. zero
*****

```

* im

The message is self-explanatory. No RUN HIGHLIGHTS are provided when Block-I input errors are encountered.

Sample Error 4. Incorrect Number of Entries in an Input Array.

Several input arrays available as input to PARTISN require a predetermined number of entries. In this example the XINTS array in the Block-II card-image input was provided with only four entries instead of the five it should have had. The error message provided by PARTISN is shown below.

```
*****
***error** fine mesh specs.ne.it
*****
```

The error message is self-explanatory.

In this case the remaining blocks of input data were successfully processed and the RUN HIGHLIGHTS are provided as shown below.

```
*****
*                                     *
*   all modules are tentatively go.   *
* **unable to write good geodst file** *
*   cross sections from cards.       *
*   interface mixing files written.   *
*   interface file asgmat written     *
*   xs files macrxs,snxedt written.   *
* *left boundary condition overridden* *
*   interface file solinp written.    *
*   edit module execution suppressed. *
*   neither editit nor edit cards exist.*
*           ** fatal input errors **  *
*                                     *
*****
```

Note that the fatal XINTS error prevented the code from creating the necessary GEODST interface file. The run was thus terminated after the remaining input data were processed.

Sample Error 5. Misplaced Array Identifier.

As discussed in the chapter “FREE FIELD INPUT REFERENCE” starting on page 5-1, arrays in the card-image input are identified by a Hollerith name or a number immediately followed by an array identifier - an equals (=) sign, a dollar (\$) sign, or an asterisk (*). The array identifier is required so that the code can recognize that the array name or number is indeed an array name or number and not an ordinary data item. In this sample a blank was inadvertently placed between the array name (ASSIGN) and the array identifier (=). The resulting message is shown below.

The first line of the message indicates that an error was found on input card 12 and is followed by the card-image column numbers. Directly below this, the card-image is reproduced. The third line indicates that the error was detected in column 32 of the card-image and the fourth line provides the self-explanatory message that a blank was found preceding the array identifier.

```
*error* card      12 *2 4 6 8(1)2 4 6 8(2)2 4 6 8(3)2 4 6 8(4)2 4 6
                  matls=matt 92235.50    assign =matls      t

*error* columns 32 - 32

**error**blank preceding an array identifier(=, $, or *)
```




COMMENTS REGARDING MULTIPLE ERRORS

As a result of the Input Module's attempt to continue processing card-image input after a fatal error has been detected, it is possible for multiple errors to be diagnosed and for multiple error messages to be printed.

When multiple error messages are printed, the user should check to see if one or more of the errors was due to a preceding error. In other words, a particular input error may cause a chain reaction of other errors. For example, suppose that the entry *IT* were inadvertently omitted from the Block-I input. The code will thus record a value of *IT*= 0. An otherwise correct entry for the *XINTS* array in Block-II, however, will now appear incorrect to the code since the code checks to see if

$$\sum_{I=1}^{IM} XINTS(I) = IT ,$$

and a message to the effect that the fine-mesh specifications (*XINTS* array) is not equal to *IT* will be printed. The user is thus advised to review multiple error messages starting with the first message printed in order to determine which errors are independent of other errors and which are results of a preceding error.



IMPLEMENTATION ERRORS

A message from the code referring to an implementation error is intended to reflect a problem with the code implementation or coding and not to reflect an input error. This will normally require the intervention of a programmer. In some cases, it means there is a memory storage problem.

As previously mentioned, an implementation error should not be caused by an input error. However, some input errors can, after the error is detected and an error message printed, ultimately cause an implementation error message before aborting. So if the implementation error message is preceded by an input error message, the implementation error message may be ignored.



WARNINGS

Warning messages are less severe than error messages and will not cause the run to terminate. They do indicate that a check in the code has found something peculiar. It thus behooves the user to look at the provided information to determine if the code was performing as the user intended. An example of such a warning that can be caused by an anomaly in the input is when the user supplies explicitly a left boundary condition for a curvilinear geometry where the implicit left boundary is reflective. The warning message will indicate that the input value was overridden.

As with error messages, warning messages from the code will occur in two different places in the printed output. Some will appear as they are detected. Some will appear in the Run Highlights where they will be found surrounded by single asterisks. Double asterisks surround error messages in the Run Highlights. Thus, the more asterisks, the worse the condition.

During the course of the problem iteration, warning messages may appear in the iteration monitor. Although succinct, these are usually self explanatory. These may or may not be due to subtle input errors of some sort. A discussion of these for each of the solvers is found in “Iteration Monitor Print” on page 3-25.

ERRORS CHAPTER INDEX

E

Error

Implementation.....	10-13
Messages.....	10-5

H

Highlights of the run	10-15
-----------------------------	-------

I

Implementation Error.....	10-13
---------------------------	-------

M

Message

Error.....	10-5
Implementation Error	10-13
Warning from Run Highlights.....	10-15

R

Run highlights.....	10-15
---------------------	-------

W

Warning messages

From the Run Highlights	10-15
-------------------------------	-------

FILE DESCRIPTIONS

Transport Methods Group, CCS-4
Los Alamos National Laboratory

11

CCS-4 — Transport Methods Group



TABLE OF CONTENTS

TABLE OF CONTENTS.....	11-3
INTRODUCTION	11-5
ASCII FILES	11-7
Problem Input File	11-7
Cross-Section Library Files	11-7
EDTOUT.....	11-7
EDTOGX	11-13
STANDARD INTERFACE FILES	11-19
CODE DEPENDENT INTERFACE FILES	11-21
AAFLXM.....	11-23
ADJMAC	11-27
AMFLUX.....	11-30
ASGMAT	11-32
AZMFLX	11-35
BXSLIB	11-37
EDITIT	11-39
FISSRC	11-46
GEODST.....	11-48
LNK3DNT	11-55
MACRXS.....	11-59
RAFLXM.....	11-63
RMFLUX.....	11-67
RZMFLX	11-69
SNXEDT	11-71
SOLINP	11-74
UCFLUX	11-86
OVERWRITTEN INPUT FILES	11-89
REFERENCES	11-91
FILES DESCRIPTIONS CHAPTER INDEX.....	11-92



INTRODUCTION

This chapter documents the detailed structure of all the files possibly used by the PARTISN package. Included are input files, output files, and interface files, all of which are files that, once created, persist after the execution of the code is completed. Scratch files, which do not persist after execution, are not included.

Some of these files are ASCII text and some are binary sequential files. ASCII text files are used for the problem input file, some cross-section files, and some output files. Binary sequential files are used for all of the interface files and some of the input and output files.

File usage history, that is, where files are created and/or where they are read, is not documented here. See “CODE PACKAGE STRUCTURE” on page 9-10 for that.

ASCII FILES

Problem Input File

The input that describes the problem to be solved is the subject of the User's Guide in this document.

Cross-Section Library Files

Cross sections may be provided in ASCII text form via the XSLIB, XSLIBB, or MACBCD files, or they may be embedded in the input file. See "COUPLED NEUTRON-GAMMA CROSS SECTIONS" on page 6-17 for a detailed description of these files.

EDTOUT

EDTOUT is a special ASCII file optionally prepared by the Edit Module of PARTISN containing geometric and edit information which can be selectively processed by the user. This description of the EDTOUT file describes the format and construct of the file. The term "section" shall be used to refer to grouped data. The term "card image" or "card" shall be used in the same context that was described on page 5-13.

1. NUMBER-OF-TITLE-RECORDS (CARDS) SECTION (Format I6)

This card contains the single number NTITLE, where NTITLE is the number of title records included in the file.

2. TITLE CARD SECTION (Format 10A8)

The title cards from the problem input are given as individual records. This section is read as follows:

```

          DIMENSION HTITLE (10, NTITLE)
          DO 10 N=1,NTITLE
          READ (NINP,20) (HTITLE(I,N), I=1,10)
10      CONTINUE
20      FORMAT(10A8)

```

3. EDIT SPECIFICATION SECTION (Format 12I6)

The edit specification section is a single card-image containing those parameters needed to process the edit-related data sections. The entries in this section are ordered as follows:

- a. IZNEZ Zone Edits present? 0/1=no/yes
- b. NZNS Number of Edit Zones
- c. IPTED Point Edits present? 0/1=no/yes
- d. NIPES Number of points for point edits. -1/0/n = all points / no points / selected points
- e. NEDISO Number of isotopes selected for edits. (Corresponds to EDISOS edits in PARTISN)
- f. MACRO Number of mixed macroscopic cross sections selected for edits. This corresponds to number of EDMATS entries PLUS the number of RESDNT edits in PARTISN.
- g. NCONS Number of constituent cross sections selected for edits. This corresponds to the number of EDCONS entries in PARTISN.
- h. NXSTYP The number of cross-section types (positions) e.g. ABS, NUSIGF, in the edits. This corresponds to the number of EDXS entries in PARTISN.
- i. NSRF The number of response functions in the edits. This corresponds to the number of RSFNAM values in PARTISN.
- j. NBG The number of Edit Broad Groups in the edits.
- k. ISADJ Adjoint problem? 0/1 = no/yes

4. GEOMETRIC SPECIFICATION SECTION (Format 12I6)

The geometric specification section contains those parameters needed to process the geometry-related data blocks. The 12 entries in this section are ordered as follows:

- a. IDIMEN The geometry-dimension of the problem 1/2/3 = one-dimensional/two-dimensional/three-dimensional
- b. IGEOM Geometry of the problem. 1=slab, 2=cylinder, etc.
- c. IM Number of coarse radial mesh intervals
- d. IT Total number of fine mesh intervals in radial direction
- e. JM Number of coarse axial mesh intervals (=1 for IDIMEN>1)
- f. JT Total number of axial fine mesh intervals (=1 for IDIMEN>1)
- g. KM Number of coarse z mesh intervals (=1 for IDIMEN<3)
- h. KT Number of fine z mesh intervals (=1 for IDIMEN<3)
- i. NDUM1 Not used

- j. NDUM2 Not used
- k. NDUM3 Not used
- l. NDUM4 Not used

ASIDE: In order to simplify the dimensions of some of the following data blocks, let us define the following internal parameters in terms of the preceding parameters found on the EDTOUT file:

- a. $NBGP1 = NBG + 1$
- b. $NXSTOT = NEDISO + MACRO + NCONS$. Note that an index I that runs from 1 to NXSTOT runs in the order indicated, namely isotopes, then mixed macroscopic, then constituents.
- c. $IMP1 = IM + 1$
- d. $JMP1 = JM + 1$
- e. $ITP1 = IT + 1$
- f. $JTP1 = JT + 1$
- g. $KMP1 = KM + 1$
- h. $KTP1 = KT + 1$
- i. NIPE = a parameter which reflects the actual number of edit points. It assumes the following value as determined by the parameter NIPES:
 $NIPE = IT*JT*KT$, if NIPES .LT. 0,
 $NIPE = 0$, if NIPES .EQ. 0,
 $NIPE = NIPES$, if NIPES .GT. 0

DATA BLOCKS

In reading the following data blocks, the user must assign data to be read to its own storage as defined by the parameters just defined. For the following data blocks we will give only a generic name for the data, the number of words in the data block, and the format-type of the block. REAL, INTEGER, and CHARACTER*8 data blocks are required. The presence or absence of a block will be indicated below by the IFF (if and only if) notation.

END ASIDE

5. GROUP ENERGY BOUNDS SECTION

ENERGY (NBGP1), 6E12

Using the data block ENERGY as an example, the block is to be read as follows:

```
        DIMENSION ENERGY(NBGP1)
        READ (NINP,400) ENERGY
400    FORMAT(6E12.5)
```

6. POINTS EDITED SECTION [IFF (NIPES .GT. 0)]
KPT (NIPE) , 12I6
7. COARSE RADIAL MESH BOUNDARIES SECTION
XMESH (IMP1) , 6E12
8. NUMBER OF FINE RADIAL MESHES PER COARSE MESH SECTION
IHX (IM) , 12I6
9. COARSE AXIAL MESH BOUNDARIES SECTION [IFF (IDIMEN .GT. 1)]
YMESH (JMP1) , 6E12
10. NUMBER OF FINE AXIAL MESHES PER COARSE MESH SECTION
[IFF (IDIMEN .GT. 1)]
IHY (JM) , 12I6
11. COARSE Z MESH BOUNDARIES SECTION [IFF (IDIMEN .GT. 2)]
ZMESH (KMP1) , 6E12
12. NUMBER OF FINE Z MESHES PER COARSE MESH SECTION
[IFF (IDIMEN .GT. 2)]
IHZ (KM) , 12I6
13. ZONE VOLUMES SECTION [IFF (NZNS .GT. 0)]
VZ (NZNS) , 6E12
14. NAMES OF ISOTOPES SECTION [IFF (NEDISO .GT. 0)]
HISO (NEDISO) , 9A8
15. NAMES OF MACROSCOPIC-EDITS SECTION [IFF (MACRO .GT. 0)]
HMACR(MACRO) , 9A8
16. NAMES OF CONSTITUENTS SECTION [IFF (NCONS .GT. 0)]
HCONS (NCONS) , 9A8
17. NAMES OF CROSS SECTION EDIT-TYPES [IFF (NXSTYP .GT. 0)]

HXSTY (NXSTYP) , 9A8

18. NAMES OF RESPONSE-FUNCTIONS SELECTED [IFF (NRSF .GT. 0)]

HRSF (NRSF) , 9A8

19. ZONES REACTION RATES SECTION [IFF (IZNED .GT. 0)]

A. CROSS-SECTION REACTION RATES SECTION
[IFF (NXSTOT*NXSTYP .GT. 0)]

XSZN (NBGP1,NZNS,NXSTYP,NXSTOT) , 6E12

This section is to be read as follows:

```

DIMENSION XSZN (NBGP1,NZNS,NXSTYP,NXSTOT)
DO 40 L=1,NXSTOT
DO 30 K=1,NXSTYP
DO 20 J=1,NZNS
READ (NINP,400) (XSZN(I,J,K,L), I=1,NBGP1)
20 CONTINUE
30 CONTINUE
40 CONTINUE
400 FORMAT(6E12.5)

```

B. RESPONSE-FUNCTION REACTION RATES SECTION [IFF (NRSF .GT. 0)]

RSZN (NBGP1,NZNS,NRSF) , 6E12

This section is to be read as follows:

```

DIMENSION RSZN (NBGP1,NZNS,NRSF)
DO 30 K=1,NRSF
DO 20 J=1,NZNS
READ (NINP,400) (RSZN(I,J,K), I=1,NBGP1)
20 CONTINUE
30 CONTINUE
400 FORMAT(6E12.5)

```

20. POINT REACTION RATES SECTION [IFF (NIPE .GT. 0)]

A. CROSS-SECTION REACTION RATES SECTION
[IFF (NXSTOT*NXSTYP .GT. 0)]

XSPT (NBGP1,NIPE,NXSTYP,NXSTOT) , 6E12

This section is to be read as follows:

```
        DIMENSION XSPT (NBGP1,NIPE,NXSTYP,NXSTOT)
        DO 40 L=1,NXSTOT
        DO 30 K=1,NXSTYP
        DO 20 J=1,NIPE
        READ (NINP,400) (XSPT(I,J,K,L), I=1,NBGP1)
20      CONTINUE
30      CONTINUE
40      CONTINUE
400     FORMAT(6E12.5)
```

B. RESPONSE-FUNCTION REACTION RATES SECTION [IFF (NRSF .GT. 0)]

RSPT (NBGP1,NIPE,NRSF) , 6E12

This section is to be read as follows:

```
        DIMENSION RSPT (NBGP1,NIPE,NRSF)
        DO 30 K=1,NRSF
        DO 20 J=1,NIPE
        READ (NINP,400) (RSPT(I,J,K), I=1,NBGP1)
20      CONTINUE
30      CONTINUE
400     FORMAT(6E12.5)
```

EDTOGX

EDTOGX is a special ASCII file optionally prepared by the Edit Module of PARTISN containing geometric, fission source, and scalar flux information which can be selectively processed by the user.

This description of the EDTOGX file describes the format and construct of the file. The term “section” shall be used to refer to grouped data. The term “card image” or “card” shall be used in the same context that was described on page 5-13.

1. NUMBER-OF-TITLE-RECORDS (CARDS) SECTION (Format I6)

This card contains the single number NTITLE, where NTITLE is the number of title cards included in the file.

2. TITLE CARD SECTION (Format 10A8)

The title cards from the problem are given as individual records. This section is read as follows

```

        DIMENSION HTITLE (10, NTITLE)
        DO 10 N=1,NTITLE
        READ (NINP,20) (HTITLE(I,N), I=1,10)
10     CONTINUE
20     FORMAT(10A8)

```

3. SPECIFICATION SECTION (Format 12I6)

The specification section is a single card-image containing those parameters needed to process the data sections. The 12 entries in this section are ordered as follows:

- a. IDIMEN The geometry dimension of the problem 1/2/3 = one-dimensional/two-dimensional/three-dimensional
- b. ISADJ Adjoint problem? , 0/1 = no/yes
- c. NGROUP Number of energy groups
- d. IM Number of radial coarse mesh intervals
- e. IT Total number of radial fine mesh intervals.
- f. JM Number of axial coarse mesh intervals (=1 for IDIMEN<2)
- g. JT Total number of axial fine mesh intervals (=1 for IDIMEN<2)
- h. KM Number of z coarse mesh intervals (=1 for IDIMEN<3)
- i. KT Total number of z fine mesh intervals. (=1 for IDIMEN<3)
- j. IFISS Fission source array present? 0/1 = no/yes

- k. IGEOM Geometry of the problem. 1=slab, 2=cylinder, etc.
- l. IADDFX Scalar fluxes present? 0/1 = yes/no

ASIDE: In order to simplify the dimensions of some of the following data blocks, the following four internal parameters are defined in terms of the preceding parameters found on the EDTOGX file:

- a. $IMP1 = IM + 1$
- b. $JMP1 = JM + 1$
- c. $KMP1 = KM + 1$
- d. $IMJMKM = IM * JM * KM$ (Total number of coarse mesh intervals)
- e. $ITJTKT = IT * JT * KT$ (Total number of fine mesh intervals)

DATA BLOCKS

In reading the following data blocks, the user must assign data to be read to its own storage as defined by the parameters just defined. For the following data blocks we will give only a generic name for the data, the number of words in the data block, and the format-type of the block. REAL, INTEGER, and CHARACTER*8 data blocks are required. The presence or absence of a block will be indicated below by the IFF (if and only if) notation.

END ASIDE

4. RADIAL DATA INFORMATION

A. FINE MESH CELL-AVERAGE-RADIUS SECTION

RDAVG (IT) , 6E12

This section is to be read as follows:

```

DIMENSION RDAVG (IT)
READ (NINP,400) RDAVG
400  FORMAT(6E12.5)

```

B. NUMBER OF RADIAL-FINE-MESHES-PER-COARSE MESH SECTION

IHX (IM) , 12I6

C. COARSE MESH RADIAL BOUNDARIES SECTION

XMESH (IMP1) , 6E12

5. AXIAL DATA INFORMATION [IFF (IDIMEN .GT. 1)]

A. FINE MESH CELL-AVERAGE-AXIAL-POSITION SECTION

ADAVG (JT) , 6E12

B. NUMBER OF AXIAL-FINE-MESHES-PER-COARSE MESH SECTION

IHY (JM) , 12I6

C. COARSE-MESH-AXIAL-BOUNDARIES SECTION

YMESH (JMP1) , 6E12

6. Z DATA INFORMATION [IFF (IDIMEN .GT. 2)]

A. FINE MESH CELL-AVERAGE-Z-POSITION SECTION

ZDAVG (KT) , 6E12

B. NUMBER OF Z-FINE-MESHES-PER-COARSE MESH SECTION

IHZ (KM) , 12I6

C. COARSE-MESH-Z-BOUNDARIES SECTION

ZMESH (KMP1) , 6E12

7. ZONE NUMBERS-BY-COARSE-MESH-INTERVAL SECTION

IDCS (IMJMKM) , 12I6

8. FISSION-SOURCE-RATE SECTION [IFF (IFISS .GT. 0)]

FISRT (ITJTKT) , 6E12

9. SCALAR FLUX SECTION [IFF (IADDFX .EQ. 0)]

FLUX (ITJTKT,NGROUP) , 6E12

This section is to be read as follows:

```
DIMENSION FLUX (ITJTKT,NGROUP)
DO 10 J=1,NGROUP
```

```
      READ (NINP,400) (FLUX(I,J) , I=1,ITJTKT)
10     CONTINUE
400    FORMAT(6E12.5)
```




STANDARD INTERFACE FILES

PARTISN makes use of interface files to transmit data between and within its modules. These interface files are binary, sequential data files.

These files are of two types, standard or code-dependent. Standard interface files are interface files whose structure and data-content formats have been standardized by the Committee on Computer Code Coordination (CCCC). Code-dependent interface files are files whose structure and data-content formats have not been standardized.

The following CCCC standard interface files are accepted, created, or otherwise used in DANTSYS: ISOTXS, GRUPXS, GEODST, NDXSRF, ZNATDN, SNCONS, FIXSRC, RTFLUX, ATFLUX, RAFLUX, AAFLUX and RZFLUX. File descriptions for these files are provided in Ref. 1.



CODE DEPENDENT INTERFACE FILES

PARTISN makes use of interface files to transmit data between and within its modules. These interface files are binary, sequential data files.

These files are of two types, standard or code-dependent. Standard interface files are interface files whose structure and data-content formats have been standardized by the Committee on Computer Code Coordination (CCCC). Code-dependent interface files are files whose structure and data-content formats have not been standardized.

The following code-dependent binary interface files are used in all of the codes in the PARTISN package: MACRXS, SNXEDT, ADJMAC, ASGMAT, SOLINP, and EDITIT.

Other code-dependent binary files provided solely as output from PARTISN, but intended to serve as interfaces to other codes or as input to subsequent runs of the same code, are the UCFLUX, BXSLIB, RMFLUX/AMFLUX, RZMFLX, RAFLXM/AAFLXM, and FISSRC files.

In this section are provided the file descriptions and a brief description of function for these code-dependent binary files. The file descriptions follow the format used for the standard interface file descriptions in Ref. 1 and Ref. 2.

AAFLXM

The AAFLXM file is a binary, code-dependent file containing the angular fluxes for the adjoint flux at each fine-mesh boundary. It differs from the standard adjoint angular flux file AAFLUX only in that the fluxes are in different order. In AAFLXM, the fluxes are in calculational order.

```
C*****-
C                                     DATE 3/03/95 -
C                                     -
CF          AAFLXM -
CE          CODE DEPENDENT COMPUTATIONAL-ORDERED ADJOINT ANGULAR FLUX -
CE          AT CELL EDGES FOR PARTISN -
C                                     -
C*****-
```

```
C-----
C                                     -
C                                     -
CN          THIS FILE PROVIDES THE EDGE ANGULAR FLUX AS ORDERED -
CN          BY THE CODE -
C                                     -
C                                     -
C-----
```

```
C-----
C                                     -
CD          ORDER OF GROUPS IS ACCORDING TO INCREASING -
CD          ENERGY. NOTE THAT DOUBLE PRECISION FLUXES ARE -
CD          GIVEN WHEN MULT=2 -
C                                     -
C-----
```

```
C-----
CS          FILE STRUCTURE -
CS          RECORD TYPE                PRESENT IF -
CS          =====                    ===== -
CS          FILE IDENTIFICATION        ALWAYS -
CS          FILE CONTROL                ALWAYS -
CS          -
CS          ***** (REPEAT FOR ALL GROUPS) -
CS          *          (GROUP NGROUP IS FIRST) -
CS          *          ***** (REPEAT FOR ALL FRONT-GOING DIRECTIONS) -
CS          *          *          COMPUTATIONAL COSINES                ALWAYS -
CS          *          *          CELL BACK-EDGE ANGULAR FLUX FOR BACK PLANE -
CS          *          ***** -
CS          *          -
CS          *          ***** (REPEAT FOR ALL Z-INTERVALS) -
CS          *          (INTERVAL NINTK IS FIRST) -
CS          *          *          ***** (REPEAT FOR ALL FRONT-GOING DIRECTIONS) -
CS          *          *          *          COMPUTATIONAL COSINES                ALWAYS -
CS          *          *          *          HORIZONTAL-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          *          VERTICAL-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          *          CELL FRONT-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          ***** -
```

```

CS * * ***** -
CS * -
CS * ***** (REPEAT FOR ALL BACK-GOING DIRECTIONS) -
CS * * COMPUTATIONAL COSINES ALWAYS -
CS * * CELL FRONT-EDGE ANGULAR FLUX FOR FRONT PLANE -
CS * ***** -
CS * -
CS * ***** (REPEAT FOR ALL Z-INTERVALS) -
CS * (INTERVAL 1 IS FIRST) -
CS * * ***** (REPEAT FOR ALL BACK-GOING DIRECTIONS) -
CS * * * COMPUTATIONAL COSINES ALWAYS -
CS * * * HORIZONTAL-EDGE ANGULAR FLUX ALWAYS -
CS * * * VERTICAL-EDGE ANGULAR FLUX ALWAYS -
CS * * * CELL BACK-EDGE ANGULAR FLUX ALWAYS -
CS * * ***** -
CS * * ***** -
CS ***** -
C -
C-----

```

```

C-----
CR FILE IDENTIFICATION -
C -
CL HNAME, (HUSE(I), I=1, 2), IVERS -
C -
CW 1+3*MULT=NUMBER OF WORDS -
C -
CD HNAME HOLLERITH FILE NAME - RAFLXM - (A6) -
CD HUSE(I) HOLLERITH USER IDENTIFICATION (A6) -
CD IVERS FILE VERSION NUMBER -
CD MULT DOUBLE PRECISION PARAMETER -
CD 1- A6 WORD IS SINGLE WORD -
CD 2- A6 WORD IS DOUBLE PRECISION WORD -
C -
C-----

```

```

C-----
CR SPECIFICATIONS (1D RECORD) -
C -
CL NDIM, NGROUP, NINTI, NINTJ, NINTK, EFFK, POWER -
C -
CW 8 =NUMBER OF WORDS -
C -
CD NDIM NUMBER OF DIMENSIONS -
CD NGROUP NUMBER OF ENERGY GROUPS -
CD NINTI NUMBER OF FIRST DIMENSION FINE MESH INTERVALS -
CD NINTJ NUMBER OF SECOND DIMENSION FINE MESH INTERVALS -
CD NINTK NUMBER OF THIRD DIMENSION FINE MESH INTERVALS. -
CD NINTK.EQ.1 IF NDIM.LE.2 -
CD EFFK EFFECTIVE MULTIPLICATION FACTOR -
CD POWER POWER IN WATTS TO WHICH FLUX IS NORMALIZED -
C -
C-----

```

```

C-----
CR COMPUTATIONAL COSINES -
C -
CL XMU, XETA, XI, WEIGHT, M, K -
C -
CW 6*MULT -
C -
CD XMU MU COSINE -

```

```

CD   XETA           ETA COSINE           -
CD   XI            XI COSINE            -
CD   WEIGHT        WEIGHT                -
CD   M             ANGLE INDEX IN THE OCTANT -
CD   K             Z-PLANE NUMBER        -
C                                         -
C-----

```

```

C-----
CR           CELL FRONT-EDGE ANGULAR FLUX -
C                                         -
CL   ((FBEDGE(I,J),I=NINTI),J=1,NINTJ) -
C                                         -
CW   NINTI*NINTJ*MULT=NUMBER OF WORDS -
C                                         -
CD   FBEDGE(I,J)   CELL FRONT-EDGE ANGULAR FLUX -
C                                         -
C-----

```

```

C-----
CR           HORIZONTAL-EDGE ANGULAR FLUX -
C                                         -
CL   ((HEDGE(I,J),I=1,NBDRYI),J=1,NINTJ) -
C                                         -
CD   HEDGE(I,J)   HORIZONTAL-EDGE-BOUNDARY ANGULAR FLUX -
C                                         -
CW   NBDRYI*NINTJ*MULT=NUMBER OF WORDS -
C                                         -
CD   NBDRYI       NINTI+1 (NUMBER OF FIRST DIMENSION FINE MESH -
CD                BOUNDARIES) -
C                                         -
C-----

```

```

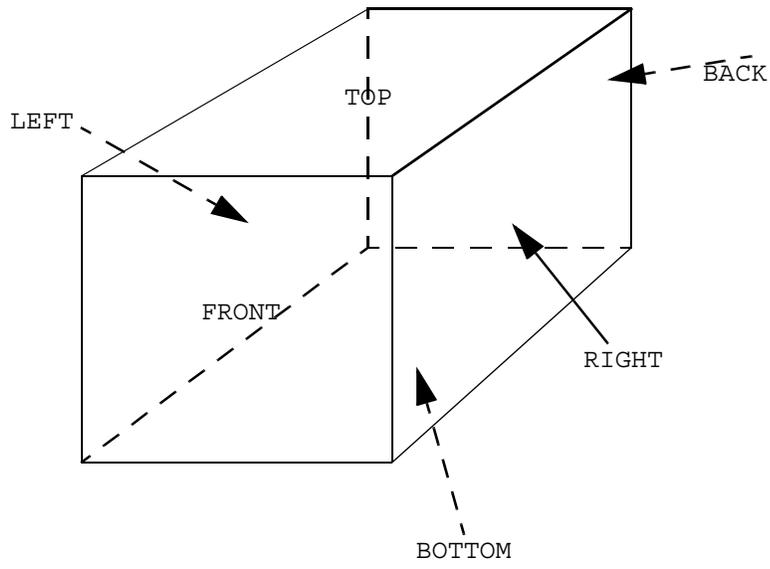
C-----
CR           VERTICAL-EDGE ANGULAR FLUX -
C                                         -
CL   ((VEDGE(I,J),I=1,NINTI),J=1,NBDRYJ) -
C                                         -
CD   VEDGE(I,J)   VERTICAL-EDGE-BOUNDARY ANGULAR FLUX -
C                                         -
CW   NINTI*NBDYJ*MULT=NUMBER OF WORDS -
C                                         -
CD   NBDRYJ       NINTJ+1 (NUMBER OF SECOND DIMENSION FINE MESH -
CD                BOUNDARIES) -
C                                         -
C-----

```

```

C-----
CR           CELL BACK-EDGE ANGULAR FLUX -
C                                         -
CL   ((FBEDGE(I,J),I=NINTI),J=1,NINTJ) -
C                                         -
CW   NINTI*NINTJ*MULT=NUMBER OF WORDS -
C                                         -
CD   FBEDGE(I,J)   CELL BACK-EDGE ANGULAR FLUX -
C                                         -
C-----

```



CEOF

ADJMAC

The ADJMAC file is the adjoint-reversed counterpart to the MACRXS interface file.

```

C*****-
C          DATE 05/12/83 -
C -
CF          ADJMAC -
CE          CODE DEPENDENT MACROSCOPIC MULTIGROUP CROSS SECTION FILE -
CE          USED IN ONEDANT SOLVER MODULE FOR ADJOINT CALCULATIONS -
C -
C*****-
C -
CN          THIS FILE PROVIDES A BASIC BROAD GROUP -
CN          LIBRARY, ORDERED BY GROUP -
C -
CN          ORDER OF GROUPS IS ACCORDING TO INCREASING ENERGY -
C -
C-----
CS          FILE STRUCTURE -
CS -
CS          RECORD TYPE                PRESENT IF -
CS          ===== -
CS          FILE IDENTIFICATION        ALWAYS -
CS          FILE CONTROL                ALWAYS -
CS          FILE DATA                  ALWAYS -
CS -
CS          ***** (REPEAT FOR ALL GROUPS) -
CS          *          PRINCIPAL CROSS SECTIONS        ALWAYS -
CS          *          SCATTERING CONTROL DATA        NORD.NE.0 -
CS          *          SCATTERING MATRIX              NORD.NE.0 -
CS          ***** -
C -
C-----
C -
C-----
CR          FILE IDENTIFICATION -
C -
CL          HNAME, (HUSE(I), I=1,2), IVERS -
C -
CW          1+3*MULT=NUMBER OF WORDS -
C -
CD          HNAME          HOLLERITH FILE NAME - ADJMAC - (A6) -
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6) -
CD          IVERS          FILE VERSION NUMBER -
CD          MULT           DOUBLE PRECISION PARAMETER -
CD          1- A6 WORD IS SINGLE WORD -
CD          2- A6 WORD IS DOUBLE PRECISION WORD -
C -
C-----
C-----
CR          FILE CONTROL -
C -
CL          NGROUP, NMAT, NORD, NED, IDPF, LNG, MAXUP, MAXDN, NPRIN, I2LP1 -
C -
CW          10=NUMBER OF WORDS -
C -
CD          NGROUP          NUMBER OF ENERGY GROUPS IN FILE -
CD          NMAT            NUMBER OF MATERIALS IN FILE -

```

```

CD   NORD          NUMBER OF LEGENDRE SCATTERING ORDERS          -
CD   NED           NUMBER OF EXTRA EDIT CROSS SECTIONS (IN ADDITION -
CD           TO THE BASIC PRINCIPAL CROSS SECTIONS)          -
CD   IDPF          0/1 NO/YES CROSS SECTION DATA ARE DOUBLE PRECISION -
CD   LNG           NUMBER OF THE LAST NEUTRON GROUP (FOR COUPLED SETS) -
CD   MAXUP         MAXIMUM NUMBER OF UPSCATTER GROUPS          -
CD   MAXDN         MAXIMUM NUMBER OF DOWNSCATTER GROUPS        -
CD   NPRIN        NUMBER OF PRINCIPAL CROSS SECTIONS          -
CD   I2LP1         0/1 = NO/YES 2L+1 TERM WAS INCLUDED IN LIBRARY -
C
C-----
C
C-----
CR           FILE DATA
C
CL   (HMAT(I) , I=1, NMAT) , (HED(J) , J=1, NEDT) , (VEL(N) , N=1, NGROUP) ,
CL   1 (EMAX(N) , N=1, NGROUP) , EMIN
C
CW   (NMAT+NEDT+2*NGROUP+1) *MULT=NUMBER OF WORDS
C
CD   HMAT(I)       HOLLERITH MATERIAL LABEL FOR MATERIAL I (A6) -
CD   HED(J)        HOLLERITH LABEL FOR J-TH CROSS SECTION POSITION (A6) -
CD   VEL(N)        MEAN NEUTRON VELOCITY IN GROUP N (CM/SEC) -
CD   EMAX(N)       MAXIMUM ENERGY BOUND OF GROUP N (EV) -
CD   EMIN          MINIMUM ENERGY BOUND OF SET (EV) -
CD   NEDT          NED+NPRIN
C
CN           THE FOUR BASIC PRINCIPAL CROSS SECTIONS
CN           ALWAYS PRESENT ARE:
CN
CN           HED(1) = 3HCHI
CN           HED(2) = 6HNUSIGF
CN           HED(3) = 5HTOTAL
CN           HED(4) = 3HABS
C
CN           ALSO PRESENT WHEN NPRIN=5 IS:
C
CN           HED(5) = 5HTRANS
C
C-----
C-----
CR           PRINCIPAL CROSS SECTIONS FOR GROUP N
C
CL   ((C(I,J) , I=1, NMAT) , J=1, NEDT)
C
CW   NMAT*NEDT*MULT=NUMBER OF WORDS
C
CD   C(I,J)        PRINCIPAL CROSS SECTIONS
C
CN           BASIC PRINCIPAL CROSS SECTIONS ALWAYS PRESENT ARE:
CN
CN           J=1  FISSION SPECTRUM
CN           J=2  FISSION NU*FISSION CROSS SECTION
CN           J=3  TOTAL CROSS SECTION
CN           J=4  ABSORPTION CROSS SECTION
C
CN           ALSO PRESENT WHEN NPRIN=5 IS:
C
CN           J=5  TRANSPORT CROSS SECTION
C
C-----
C-----
CR           SCATTERING CONTROL BLOCK FOR GROUP N

```

```

C -
CC      PRESENT IF NORD.GT.0 -
C -
CL      ((NGPB(L,J),L=1,NORD),J=1,NMAT) -
CL      ((IFSG(L,J),L=1,NORD),J=1,NMAT) -
C -
CW      2*NORD*NMAT=NUMBER OF WORDS -
C -
CD      NGPB(L,J)   NUMBER OF SOURCE GROUPS THAT CAN SCATTER INTO GROUP N -
CD      IFSG(L,J)   GROUP NUMBER OF THE FIRST SOURCE GROUP -
CD      L           LEGENDRE ORDER NUMBER -
CD      J           MATERIAL NUMBER -
C -
C-----
C-----
CR      SCATTERING SUB-BLOCK FOR GROUP N -
C -
CC      PRESENT IF NORD.GT.0 -
C -
CL      (SCAT(I),I=1,NTAB) -
C -
CW      NTAB*MULT=NUMBER OF WORDS -
C -
CD      SCAT(I)     SCATTERING CROSS SECTION -
C -
CD      NTAB        TABLE LENGTH OF THE CROSS SECTIONS FOR SCATTERING -
CD                  INTO GROUP N. THIS IS FOR ALL MATERIALS AND ALL -
CD                  LEGENDRE ORDERS, THUS IT IS THE SUM OF NGPB(L,J) -
CD                  OVER L FROM 1 TO NORD AND OVER J FROM 1 TO NMAT. -
C -
CN      THE SCATTERING CROSS SECTIONS ARE PACKED IN BANDS, -
CN      ONE FOR EACH LEGENDRE ORDER AND MATERIAL. EACH BAND -
CN      CONTAINS THE NGPB GROUPS WHICH SCATTER INTO GROUP -
CN      N. THE FIRST SOURCE GROUP NUMBER IS IFSG AND -
CN      THE LAST IS IFSG-NGPB+1. THE NORD BANDS FOR THE -
CN      FIRST MATERIAL APPEAR FIRST (P0, P1, ...) FOLLOWED -
CN      BY THE NORD BANDS FOR THE SECOND, ETC. -
C -
CN      HIGHER LEGENDRE ORDER SCATTERING CROSS SECTIONS -
CN      INCLUDE A 2*L+1 FACTOR WHERE L IS THE LEGENDRE -
CN      ORDER. -
C -
C-----
CEOF

```

AMFLUX

The AMFLUX code-dependent file contains, in binary form, the spherical harmonics adjoint angular flux moments for all spatial fine mesh points and all energy groups. It is optionally produced by the Solver Module.

```

C*****
C          DATE 04/01/85
C
C          AMFLUX-IV
CE          ADJOINT FLUX MOMENTS
C
C*****
C          ORDER OF GROUPS IS ACCORDING TO INCREASING ENERGY
C
C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE                PRESENT IF
CS          =====
CS          FILE IDENTIFICATION        ALWAYS
CS          SPECIFICATIONS              ALWAYS
CS
CS          ***** (REPEAT FOR ALL GROUPS)
CS          *          ADJOINT MOMENTS FLUXES          ALWAYS
CS          *****
C
C-----
CR          FILE IDENTIFICATION
C
CL          HNAME, (HUSE(I), I=1,2), IVERS
C
CW          1+3*MULT=NUMBER OF WORDS
C
CD          HNAME          HOLLERITH FILE NAME - RMFLUX - (A6)
CD          HNAME          HOLLERITH FILE NAME - (A6)
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)
CD          IVERS          FILE VERSION NUMBER
CD          MULT           DOUBLE PRECISION PARAMETER
CD                        1- A6 WORD IS SINGLE WORD
CD                        2- A6 WORD IS DOUBLE PRECISION WORD
C
C-----
CR          SPECIFICATIONS          (1D RECORD)
C
CL          NDIM, NGROUP, NINTI, NINTJ, NINTK, NORD, EFFK, POWER, OITNO
C
CW          9=NUMBER OF WORDS
C
CD          NDIM           NUMBER OF DIMENSIONS
CD          NGROUP         NUMBER OF GROUPS
CD          NINTI          NUMBER OF FIRST DIMENSION INTERVALS
CD          NINTJ          NUMBER OF SECOND DIMENSION INTERVALS
CD                        NINTJ.EQ.1 IF NDIM.EQ.1
CD          NINTK         NUMBER OF THIRD DIMENSION INTERVALS

```

```

CD          NINTK.EQ.1 IF NDIM.LE.2 -
CD  NORD    NUMBER OF LEGENDRE MOMENTS -
CD  EFFK    EFFECTIVE MULTIPLICATION FACTOR -
CD  POWER   POWER IN WATTS TO WHICH FLUX IS NORMALIZED -
CD  OITNO   OUTER ITERATION NUMBER -
C-----
C -
C-----
CR          ADJOINT MOMENTS FLUXES ON MULTIDIMENSIONAL INTERVALS -
C          (2D RECORD) -
C -
CL  ((FLUX(M,I),M=1,NORD),I=1,NINTI)  ----NOTE STRUCTURE BELOW--- -
C -
CW  NORD*NINTI=NUMBER OF WORDS -
C -
C  DO 1 K=1,NINTK -
C  DO 1 J=1,NINTJ -
C  1 READ (N) *LIST AS ABOVE* -
C -
CD  FLUX(M,I)  ADJOINT FLUX MOMENTS ON FIRST DIMENSION -
CD            INTERVALS. -
C -
C-----
CEOF -

```

ASGMAT

The ASGMAT interface file contains the information needed by the Solver and Edit Modules to assign materials to zones to create the zone macroscopic cross sections.

```

C*****-
C          DATE 09/18/81-
C-
CF          ASGMAT-
CE          CODE DEPENDENT FILE ASSIGNING MATERIALS TO ZONES-
C-
C-
C*****-
C-
CN          THIS FILE CONTAINS THE INFORMATION FROM THE INPUT-
CN          ARRAYS ASSIGN= AND ASGMOD=-
C-
C-----
CS          FILE STRUCTURE-
CS-
CS          RECORD TYPE          PRESENT IF-
CS          =====
CS          FILE IDENTIFICATION  ALWAYS-
CS          FILE CONTROL         ALWAYS-
CS          COMPATABILITY CODE   ALWAYS-
CS          MATERIAL NAMES       ALWAYS-
CS          ZONE NAMES           ALWAYS-
CS          NUMBER OF MATERIALS PER ZONE  MPZTOT.NE.0-
CS          MATERIAL LIST FOR ALL ZONES  MPZTOT.NE.0-
CS          MATERIAL CONCENTRATIONS     MPZTOT.NE.0-
CS          MATERIAL CONCENTRATION FACTORS MPZTOT.NE.0-
CS          CONCENTRATION MODIFIER      MPZTOT.NE.0-
C-
C-----
C-----
CR          FILE IDENTIFICATION-
C-
CL          HNAME, (HUSE(I), I=1,2), IVERS-
C-
CW          1+3*MULT=NUMBER OF WORDS-
C-
CD          HNAME          HOLLERITH FILE NAME - ASGMAT - (A6)-
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)-
CD          IVERS          FILE VERSION NUMBER-
CD          MULT           DOUBLE PRECISION PARAMETER-
CD          1- A6 WORD IS SINGLE WORD-
CD          2- A6 WORD IS DOUBLE PRECISION WORD-
C-
C-----
C-----
CR          FILE CONTROL-
C-
CL          MT, NZONE, MPZTOT, FMMIX-
C-
CW          4=NUMBER OF WORDS-
C-
CD          1  MT          NUMBER OF MATERIALS-
CD          2  NZONE       NUMBER OF ZONES-
CD          3  MPZTOT      IN-SOLVER MIXING TABLE LENGTH-

```

```

CD   4  FMMIX      0/1 NO/YES VOL FRACTION MIXING BY FINE MESH      -
C                                           -
C-----
C                                           -
C-----
CR           COMPATABILITY CODE WORDS                                -
C                                           -
CL   CODE1, CODE2                                                    -
C                                           -
CW   2*MULT=NUMBER OF WORDS                                         -
C                                           -
CD   1  CODE1      DATE OF THE MACRXS FILE TO WHICH ASGMAT APPLIES  -
CD   2  CODE2      TIME OF THE MACRXS FILE TO WHICH ASGMAT APPLIES  -
C                                           -
C-----
C                                           -
C-----
CR           MATERIAL NAMES                                          -
C                                           -
CL   (MATNAM(I) , I=1, MT)                                           -
C                                           -
CW   MT*MULT=NUMBER OF WORDS                                         -
C                                           -
CD   MATNAM(I) HOLLERITH NAME FOR THE I-TH MATERIAL                 -
C                                           -
C-----
C                                           -
C-----
CR           ZONE NAMES                                              -
C                                           -
CL   (ZONNAM(I) , I=1, NZONE)                                         -
C                                           -
CW   NZONE*MULT=NUMBER OF WORDS                                       -
C                                           -
CD   ZONNAM(I) HOLLERITH NAME FOR THE I-TH ZONE                     -
C                                           -
C-----
C                                           -
C-----
CR           NUMBER OF MATERIALS PER ZONE                             -
C                                           -
CC   PRESENT IF MPZTOT.NE.0                                           -
C                                           -
CL   (NUMZON(I) , I=1, NZONE)                                         -
C                                           -
CW   NZONE=NUMBER OF WORDS                                           -
C                                           -
CD   NUMZON(I) NUMBER OF MATERIALS IN THE I-TH ZONE                 -
C                                           -
C-----
C                                           -
C-----
CR           MATERIAL LIST FOR ALL ZONES                              -
C                                           -
CC   PRESENT IF MPZTOT.NE.0                                           -
C                                           -
CL   (MATLST(I) , I=1, MPZTOT)                                         -
C                                           -
CW   MPZTOT=NUMBER OF WORDS                                           -
C                                           -
CD   MATLST      PACKED LIST OF MATERIAL NUMBERS. MATERIALS FOR ZONE 1-
CD               FOLLOWED BY THOSE FOR ZONE 2, THEN ZONE3, ETC.      -
C                                           -
C-----
C                                           -
C

```

```

C-----
CR          MATERIAL CONCENTRATIONS                      -
C                                                  -
CC          PRESENT IF MPZTOT.NE.0                      -
C                                                  -
CL          (CONC(I), I=1, MPZTOT)                      -
C                                                  -
CW          MPZTOT*MULT=NUMBER OF WORDS                 -
C                                                  -
CD          CONC(I)   CONCENTRATION OF THE I-TH MATERIAL IN THE MATLST -
CD                      ARRAY                          -
C                                                  -
C-----
C
C-----
CR          MATERIAL CONCENTRATION FACTORS                -
C                                                  -
CC          PRESENT IF MPZTOT.NE.0                      -
C                                                  -
CL          (C1(I), I=1, MPZTOT)                        -
C                                                  -
CW          MPZTOT*MULT=NUMBER OF WORDS                 -
C                                                  -
CD          C1(I)    CONCENTRATION FACTOR FOR THE I-TH MATERIAL IN THE -
CD                      MATLST ARRAY                    -
C                                                  -
C-----
C
C-----
CR          CONCENTRATION MODIFIER                        -
C                                                  -
CC          PRESENT IF MPZTOT.NE.0                      -
C                                                  -
CL          CMOD                                         -
C                                                  -
CW          1*MULT=NUMBER OF WORDS                      -
C                                                  -
CD          CMOD    INPUT VALUE OF THE CONCENTRATION MODIFIER -
C                                                  -
C-----
CEOF

```

AZMFLX

The AZMFLX file is a binary, code-dependent file containing the spherical harmonics adjoint angular flux moments averaged over each zone for each energy group. The zones over which the fluxes are averaged are the zones used in the Solver Module and not the Edit Zones optionally used in the Edit Module.

```

C*****
C                                DATE 01/28/95                                -
C                                                                -
CF          AZMFLX-IV                                                    -
CE          ADJOINT ZONE AVERAGED FLUX MOMENTS BY GROUP                -
C                                                                -
C*****
C                                                                -
CN                                ORDER OF GROUPS IS ACCORDING TO INCREASING ENERGY -
C                                                                -
C-----
CS          FILE STRUCTURE                                              -
CS                                                                -
CS          RECORD TYPE                                                PRESENT IF                          -
CS          =====
CS          FILE IDENTIFICATION                                        ALWAYS                          -
CS          SPECIFICATIONS                                          ALWAYS                          -
CS                                                                -
CS          ***** (REPEAT FOR ALL GROUPS)                            -
CS          *          ZONE AVERAGED FLUX MOMENTS                      ALWAYS                          -
CS          *****
C                                                                -
C-----
C-----
CR          FILE IDENTIFICATION                                          -
C                                                                -
CL          HNAME, (HUSE(I), I=1, 2), IVERS                            -
C                                                                -
CW          1+3*MULT=NUMBER OF WORDS                                    -
C                                                                -
CD          HNAME              HOLLERITH FILE NAME - RZMFLX - (A6)    -
CD          HNAME              HOLLERITH FILE NAME - (A6)            -
CD          HUSE(I)            HOLLERITH USER IDENTIFICATION (A6)    -
CD          IVERS              FILE VERSION NUMBER                    -
CD          MULT                DOUBLE PRECISION PARAMETER            -
CD                                1- A6 WORD IS SINGLE WORD            -
CD                                2- A6 WORD IS DOUBLE PRECISION WORD  -
C                                                                -
C-----
C-----
CR          SPECIFICATIONS          (1D RECORD)                            -
C                                                                -
CL          NDIM, NGROUP, NZONE, DUM, DUM, NORD, EFFK, POWER, OITNO    -
C                                                                -
CW          9=NUMBER OF WORDS                                           -
C                                                                -
CD          NDIM                NUMBER OF DIMENSIONS                    -
CD          NGROUP              NUMBER OF GROUPS                        -
CD          NZONE                NUMBER OF GEOMETRIC ZONES              -
CD          DUM                  DUMMY, NOT USED                        -

```

CD	DUM	DUMMY, NOT USED	-
CD	NORD	NUMBER OF LEGENDRE MOMENTS	-
CD	EFFK	EFFECTIVE MULTIPLICATION FACTOR	-
CD	POWER	POWER IN WATTS TO WHICH FLUX IS NORMALIZED	-
CD	OITNO	OUTER ITERATION NUMBER	-
C			-
C	-----		-
C			-
C	-----		-
CR	ADJOINT FLUX MOMENTS AVERAGED OVER EACH ZONE		-
C	(2D RECORD)		-
C			-
CL	((FLUX(M, I), M=1, NORD), I=1, NZONE)		-
C			-
CW	NORD*NZONE=NUMBER OF WORDS		-
C			-
C			-
CD	FLUX(M, I)	ADJOINT FLUX MOMENT AVERAGES FOR EACH	-
CD		ZONE.	-
C			-
C	-----		-
CEOF			-

BXSLIB

The BXSLIB code-dependent file contains, in binary form, the cross sections and other cross section and mixing information as described in “Binary Form of Card-Image Libraries (the BXSLIB file)” on page 6-12.

```

C*****
C          DATE 09/22/88
C
CF          BXSLIB
CE          MICROSCOPIC GROUP NEUTRON CROSS SECTIONS FROM CARDS
C
CN          THIS FILE CONTAINS IN BINARY FORM THE
CN          BLOCK III ONEDANT INPUT TOGETHER WITH THE
CN          CROSS SECTIONS FROM THE ORIGINAL CARD LIBRARY.
CN          THE FILE ALSO MAY CONTAIN ISOTOPE/ATOMIC WEIGHT-
CN          PAIRS FROM THE BLOCK IV ONEDANT INPUT
C
C*****
C-----
CS          FILE STRUCTURE
C
CS          RECORD TYPE                                PRESENT IF
CS          =====                                =====
CS          FILE IDENTIFICATION                        ALWAYS
CS          FILE CONTROL                               ALWAYS
CS          FILE DATA                                 ALWAYS
C
CS          ***** (REPEAT FOR ALL ISOTOPES)
CS          * ***** (REPEAT FOR ALL LEGENDRE ORDERS)
CS          * *      CROSS SECTION SET                ALWAYS
CS          * *****
CS          *****
CS          ISOTOPE LABEL/ATOMIC WEIGHT PAIR          NISOAW.GT.0
C
C-----
C-----
CR          FILE IDENTIFICATION
C
CL          HNAME, (HUSE(I), I=1,2), IVERS
C
CW          1+3*MULT=NUMBER OF WORDS
C
C
CD          HNAME          HOLLERITH FILE NAME - BXSLIB - (A6)
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)
CD          IVERS          FILE VERSION NUMBER
CD          MULT           DOUBLE PRECISION PARAMETER
CD                          1- A6 WORD IS SINGLE WORD
CD                          2- A6 WORD IS DOUBLE PRECISION WORD
C
C-----
C-----
CR          FILE CONTROL
C
CL          NGROUP, NISO, IHM, IHT, IHS, MAXT, NISOAW, NXSREC, I2LP1
C
CW          9=NUMBER OF WORDS
C
C

```

```

CD   NGROUP      NUMBER OF ENERGY GROUPS IN FILE      -
CD   NISO        NUMBER OF ISOTOPES IN FILE        -
CD   IHM         TABLE LENGTH (NUMBER OF CROSS SECTIONS FOR
CD              ONE ISOTOPE, FOR ONE GROUP, AND FOR ONE LEGENDRE
CD              ORDER)                             -
CD   IHT         TOTAL CROSS SECTION POSITION IN THE TABLE -
CD   IHS         SELF SCATTER CROSS SECTION POSITION -
CD   MAXT        MAXIMUM NUMBER OF LEGENDRE MOMENTS -
CD   NISOAW      NUMBER OF ISOTOPE LABEL/ATOMIC WEIGHT PAIR -
CD   NXSREC      TOTAL NUMBER OF CROSS SECTION SETS -
CD   I2LP1       0/1 - NO/YES SCATTERING CROSS SECTIONS CONTAIN -
CD              2L+1 FACTOR                        -
C                                           -
C-----
CR          FILE DATA                          -
C                                           -
CL   (HSETID(I), I=1, 12), (HISONM(I), I=1, NISO), (EDNAME(I), I=1, IHT-3), -
CL   1 (CHI(J), J=1, NGROUP), (VEL(J), J=1, NGROUP), -
CL   2 (EMAX(J), J=1, NGROUP), EMIN, (NSPI(I), I=1, NISO) -
C                                           -
CW   (NISO+12+IHT-3)*MULT+(3*NGROUP+1)*MULT+NISO=NUMBER OF WORDS -
C                                           -
C                                           -
CD   HSETID(I)   HOLLERITH IDENTIFICATION OF FILE (A6) -
CD   HISONM(I)   HOLLERITH ISOTOPE LABEL FOR ISOTOPE I (A6) -
CD   EDNAME(I)   HOLLERITH NAME FOR EDIT POSITIONS -
CD              PRECEDING SIGMA ABSORPTION FOR POSITION I (A6) -
CD   CHI(J)      FILE-WIDE FISSION SPECTRUM(ZEROES IN ONEDANT) -
CD   VEL(J)      MEAN NEUTRON VELOCITY IN GROUP J (CM/SEC) -
CD   EMAX(J)     MAXIMUM ENERGY BOUND OF GROUP J (EV) -
CD   EMIN        MINIMUM ENERGY BOUND OF SET (EV) -
CD   NSPI(I)     NUMBER OF LEGENDRE ORDERS FOR ISOTOPE I -
C                                           -
C-----
CR          CROSS SECTION SET FOR ISOTOPE I AND LEGENDRE ORDER M -
C                                           -
CL   ((C(I,J), I=1, IHM), J=1, NGROUP) -
C                                           -
CW   IHM*NGROUP*MULT=NUMBER OF WORDS -
C                                           -
C-----
CR          HOLLERITH ISOTOPE LABEL/ATOMIC WEIGHT PAIR SET -
C                                           -
CC   PRESENT IF NISOAW.GT.0 -
C                                           -
CL   (ATWTP(I), I=1, 2*NISOAW) -
C                                           -
CW   2*NISOAW*MULT=NUMBER OF WORDS -
C                                           -
C-----
CEOF

```

EDITIT

The EDITIT code-dependent interface file contains information specific to the Edit Module, mainly information from Block VI of the card-image input.

```

C*****
C                                DATE 01/28/95
C
CF          EDITIT
CE          CODE DEPENDENT FILE OF INFORMATION SPECIFIC TO THE
CE          ONEDANT EDIT MODULE
C
C
C*****
C
CN          THIS FILE CONTAINS THE CARD INPUT INFORMATION
CN          FROM BLOCK VI
C
C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE                                PRESENT IF
CS          =====
CS          FILE IDENTIFICATION                        ALWAYS
CS          RAW CONTROLS AND DIMENSIONS                ALWAYS
CS          DEFAULTED CONTROLS AND DIMENSIONS         ALWAYS
CS          RAW FLOATING INPUT DATA                  ALWAYS
CS          FINE GROUPS PER BROAD GROUP               ALWAYS
CS          ZONE NUMBERS                               NZNS.NE.0
CS          POINT NUMBERS                             NIPE.NE.0
CS          IPLANE NUMBERS                            NIPLNE.NE.0
CS          JPLANE NUMBERS                            NJPLNE.NE.0
CS          KPLANE NUMBERS                            NKPLNE.NE.0
CS          CROSS SECTION POSITIONS                   NPOS.NE.0
CS          ISOTOPE NUMBERS TO EDIT                   NISO.NE.0
CS          MATERIAL NUMBERS TO EDIT                  MACRO.NE.0
CS          CONSTITUENT NUMBERS TO EDIT               NCONS.NE.0
CS          RESPONSE FUNCTION NAMES                   IDOSE.NE.0
CS          ***** (REPEAT FOR ALL RESPONSE FUNCTIONS)
CS          *     RESPONSE FUNCTION ENERGY VECTOR    IDOSE.NE.0
CS          *     RESPONSE FUNCTION X SPATIAL VECTOR  IDOSE.NE.0
CS          *     RESPONSE FUNCT Y SPATIAL VECTOR     IDOSE.NE.0.AND.IDIMEN.GT.1
CS          *     RESPONSE FUNCT Z SPATIAL VECTOR     IDOSE.NE.0.AND.IDIMEN.GT.2
CS          *****
CS          CROSS SECTION SUMMING ARRAY -             IXSUM.NE.0
CS          RESPONSE FUNCTION SUMMING ARRAY           IRSUM.NE.0
CS          FINE X MESH DENSITY FACTORS               IDEN.NE.0
CS          FINE Y MESH DENSITY FACTORS               IDEN.NE.0 .AND. IDIMEN.GT.1
CS          FINE Z MESH DENSITY FACTORS               IDEN.NE.0 .AND. IDIMEN.GT.2
C
C-----
C
C-----
CR          FILE IDENTIFICATION
C
CL          HNAME, (HUSE(I), I=1,2), IVERS
C
CW          1+3*MULT=NUMBER OF WORDS
C
CD          HNAME          HOLLERITH FILE NAME - EDITIT - (A6)

```

```

CD   HUSE(I)           HOLLERITH USER IDENTIFICATION (A6)      -
CD   IVERS            FILE VERSION NUMBER                    -
CD   MULT              DOUBLE PRECISION PARAMETER            -
CD                       1- A6 WORD IS SINGLE WORD           -
CD                       2- A6 WORD IS DOUBLE PRECISION WORD  -
C-----
C
C-----
CR           RAW CONTROLS AND DIMENSIONS                      -
C
CL   IEDOPT,PTED, NIPE, IKND, ZNED, NZNS,IXSUM,IRSUM, NPOS, NISO, -
CL 1 NCONS, MACRO, IDOSE, IGRPED, LNG, NBG, IDEN, NGROUP, AJED, -
CL 2 ITED, JTED, KTED, IRZFLX, IRZMFX, EDOUTF, IFLUX1, IPRPLT, -
CL 3 NIPLNE, NJPLNE, NKPLNE                                  -
C
CW   75=NUMBER OF WORDS                                     -
C
CD 1 IEDOPT NOT USED                                         -
CD 2 PTED 0/1 - NO/YES DO POINT EDIT                         -
CD 3 NIPE NUMBER OF POINTS TO EDIT                           -
CD 4 BYVOLP 0/1 - NO/YES MULTIPLY REACTION RATES BY MESH VOLUME -
CD 5 ZNED 0/1 - NO/YES DO ZONE EDIT                          -
CD 6 NZNS NUMBER OF EDIT ZONES                               -
CD 7 IXSUM LENGTH OF CROSS SECTION SUMMING TABLE           -
CD 8 IRSUM LENGTH OF RESPONSE FUNCTION SUMMING TABLE       -
CD 9 NPOS NUMBER OF CROSS SECTION POSITIONS TO EDIT         -
CD 10 NISO NUMBER OF ISOTOPES TO EDIT                        -
C
CD 11 NCONS NUMBER OF ISOTOPES TO EDIT AS CONSTITUENTS     -
CD 12 MACRO NUMBER OF MATERIALS TO EDIT                      -
CD 13 IDOSE NUMBER OF RESPONSE FUNCTIONS TO EDIT             -
CD 14 IGRPED 0/1/2/3 - ENERGY GROUP PRINT OPTIONS          -
CD 15 LNG NUMBER OF THE LAST NEUTRON GROUP                   -
CD 16 NBG NUMBER OF BROAD ENERGY GROUPS                     -
CD 17 IDEN 0/1 - NO/YES THERE ARE FINE MESH DENSITY FACTORS -
CD 18 NGROUP NUMBER OF FINE ENERGY GROUPS                   -
CD 19 AJED 0/1 - NO/YES THIS IS AN ADJOINT EDIT              -
CD 20 ITED NUMBER OF FINE RADIAL MESH                         -
CD 21 JTED NUMBER OF FINE AXIAL MESH                          -
CD 22 KTED NUMBER OF FINE Z DIRECTION MESH                   -
CD 23 IRZFLX 0/1 - NO/YES WRITE CCCC RZFLUX FILE             -
CD 24 IRZMFX 0/1 - NO/YES WRITE ZONE MOMENTS FILE            -
CD 25 EDOUTF -3/-2/0/1/2/3 ASC EDIT FILE PREPARATION INDICATOR -
CD 26 IFLUX1 0/1 - NO/YES MAKE ALL INPUT FLUXES UNITY       -
CD 27 IPRPLT 0/1/2/3 PRINT/NOTHING/TECPLT/BOTH VISUALIZATION OPT -
CD 28 NIPLNE NUMBER OF I PLANES PLOTTED                      -
CD 29 NJPLNE NUMBER OF J PLANES PLOTTED                      -
CD 30 NKPLNE NUMBER OF K PLANES PLOTTED                      -
C-----
C
C-----
CR           DEFAULTED CONTROLS AND DIMENSIONS                -
C
CN   THIS RECORD IS THE SAME FORMAT AS THE RAW CONTROLS AND -
CN   DIMENSION RECORD ABOVE, BUT IT CONTAINS THE DEFAULTED -
CN   VALUES FOR EACH VARIABLE                               -
C-----
C
C-----
CR           RAW FLOATING INPUT DATA                          -
C
CL   POWER, MEVPER                                           -

```

```

C -
CW 2*MULT=NUMBER OF WORDS -
C -
CD 1 POWER NORMALIZE TO POWER -
CD 2 MEVPER MEV RELEASED PER FISSION -
C -
C-----
C -
C-----
CR FINE GROUPS PER BROAD GROUP -
C -
CL (ICOLL(G),G=1,NBG) -
C -
CW NBG=NUMBER OF WORDS -
C -
CD ICOLL(G) NUMBER OF FINE GROUPS IN BROAD GROUP G -
C -
C-----
C -
C-----
CR ZONE NUMBERS -
C -
CC PRESENT IF NZNS.GT.0 -
C -
CL (EDZONE(I),I=1,IT*JT*KT) -
C -
CW IT*JT*KT=NUMBER OF WORDS -
C -
CD EDZONE(I) EDIT ZONE NUMBER FOR THE I-TH FINE MESH -
C -
C-----
C -
C-----
CR POINTS TO EDIT -
C -
CC PRESENT IF NIPE.GT.0 -
C -
CL (POINTS(I),I=1,NIPE) -
C -
CW NIPE=NUMBER OF WORDS -
C -
CD POINTS(I) NUMBER OF THE I-TH POINT TO EDIT -
C -
C-----
C -
C-----
CR IPLANES TO EDIT -
C -
CC PRESENT IF NIPLNE.GT.0 -
C -
CL (IPLNES(I),I=1,NIPLNE) -
C -
CW NIPLNE=NUMBER OF WORDS -
C -
CD IPLNES(I) NUMBER OF THE I-TH PLANE TO EDIT -
C -
C-----
C -
C-----
CR JPLANES TO EDIT -
C -
CC PRESENT IF NJPLNE.GT.0 -
C -
CL (JPLNES(I),I=1,NJPLNE) -
C -
CW NJPLNE=NUMBER OF WORDS -

```

```

C
CD      JPLNES(I) NUMBER OF THE J-TH PLANE TO EDIT
C
C-----
C
C-----
CR      KPLANES TO EDIT
C
CC      PRESENT IF NKIPLNE.GT.0
C
CL      (KPLNES(I), I=1, NKPLNE)
C
CW      NKPLNE=NUMBER OF WORDS
C
CD      KPLNES(I) NUMBER OF THE K-TH PLANE TO EDIT
C
C-----
C
C-----
CR      CROSS SECTION POSITIONS TO EDIT
C
CC      PRESENT IF NPOS.GT.0
C
CL      (EDXS(I), I=1, NPOS)
C
CW      NPOS=NUMBER OF WORDS
C
CD      EDXS(I) POSITION NUMBER TO EDIT(IN NUMERIC FORM)
C
C-----
C
C-----
CR      ISOTOPE NUMBERS TO EDIT
C
CC      PRESENT IF NISO.GT.0
C
CL      (EDISOS(I), I=1, NISO)
C
CW      NISO=NUMBER OF WORDS
C
CD      EDISOS(I) ISOTOPE NUMBER TO EDIT(IN NUMERIC FORM)
C
C-----
C
C-----
CR      MATERIAL NUMBERS TO EDIT
C
CC      PRESENT IF MACRO.GT.0
C
CL      (EDMATS(I), I=1, MACRO)
C
CW      MACRO=NUMBER OF WORDS
C
CD      EDMATS(I) MATERIAL NUMBER TO EDIT(IN NUMERIC FORM)
C
C-----
C
C-----
CR      CONSTITUENT NUMBERS TO EDIT
C
CC      PRESENT IF NCONS.GT.0
C
CL      (EDCONS(I), I=1, NCONS)
C

```

```

CW   NCONS=NUMBER OF WORDS -
C -
CD   EDCONS(I) CONSTITUENT NUMBER TO EDIT(IN NUMERIC FORM) -
C -
C-----
C -
C-----
CR   RESPONSE FUNCTION NAMES -
C -
CC   PRESENT IF IDOSE.GT.0 -
C -
CL   (RSFNAM(I), I=1, IDOSE) -
C -
CW   IDOSE*MULT=NUMBER OF WORDS -
C -
CD   RSFNAM(I) HOLLERITH NAME FOR THE I-TH RESPONSE FUNCTION -
C -
C-----
C -
C-----
CR   RESPONSE FUNCTION ENERGY VECTOR -
C -
CC   PRESENT IF IDOSE.GT.0 -
C -
CL   (RSFE(I), I=1, NGROUP) -
C -
CW   NGROUP*MULT=NUMBER OF WORDS -
C -
CD   RSFE(I) RESPONSE FOR GROUP I -
C -
C-----
C -
C-----
CR   RESPONSE FUNCTION SPATIAL VECTOR IN X -
C -
CC   PRESENT IF IDOSE.LT.0 -
C -
CL   (RSFX(I), I=1, IT) -
C -
CW   IT*MULT=NUMBER OF WORDS -
C -
CD   RSFX(I) RESPONSE FUNCTION FOR FINE MESH I -
C -
C-----
C -
C-----
CR   RESPONSE FUNCTION SPATIAL VECTOR IN Y -
C -
CC   PRESENT IF IDOSE.LT.0 .AND. IDIMEN.GT.1 -
C -
CL   (RSFY(J), J=1, JT) -
C -
CW   JT*MULT=NUMBER OF WORDS -
C -
CD   RSFY(J) RESPONSE FUNCTION FOR FINE MESH J -
C -
C-----
C -
C-----
CR   RESPONSE FUNCTION SPATIAL VECTOR IN Z -
C -
CC   PRESENT IF IDOSE.LT.0 .AND. IDIMEN.GT.2 -
C -
CL   (RSFZ(K), K=1, KT) -
C -

```

```

CW      KT*MULT=NUMBER OF WORDS          -
C                                             -
CD      RSFZ(K)   RESPONSE FUNCTION FOR FINE MESH K  -
C                                             -
C-----
C-----
CR      CROSS SECTION SUMMING ARRAY          -
C                                             -
CC      PRESENT IF IXSUM.GT.0              -
C                                             -
CL      (MICSUM(I) , I=1, IXSUM)          -
C                                             -
CW      IXSUM=NUMBER OF WORDS              -
C                                             -
CD      MICSUM    INPUT SUMMING ARRAY IN NUMERIC FORM  -
C                                             -
C-----
C-----
CR      RESPONSE FUNCTION SUMMING ARRAY          -
C                                             -
CC      PRESENT IF IRSUM.GT.0              -
C                                             -
CL      (IRSUMS(I) , I=1, IRSUM)          -
C                                             -
CW      IRSUM=NUMBER OF WORDS              -
C                                             -
CD      IRSUMS    INPUT SUMMING ARRAY IN NUMERIC FORM  -
C                                             -
C-----
C-----
CR      FINE MESH DENSITY VECTOR IN X          -
C                                             -
CC      PRESENT IF IDEN.GT.0              -
C                                             -
CL      (XDF(I) , I=1, IT)                -
C                                             -
CW      IT*MULT=NUMBER OF WORDS              -
C                                             -
CD      XDF(I)   DENSITY FACTOR FOR THE I-TH FINE MESH  -
C                                             -
C-----
C-----
CR      FINE MESH DENSITY VECTOR IN Y          -
C                                             -
CC      PRESENT IF IDEN.GT.0 .AND. IDIMEN.GT.1  -
C                                             -
CL      (YDF(J) , J=1, JT)                -
C                                             -
CW      JT*MULT=NUMBER OF WORDS              -
C                                             -
CD      YDF(J)   DENSITY FACTOR FOR THE J-TH FINE MESH  -
C                                             -
C-----
C-----
CR      FINE MESH DENSITY VECTOR IN Z          -
C                                             -
CC      PRESENT IF IDEN.GT.0 .AND. IDIMEN.GT.2  -
C                                             -
CL      (ZDF(K) , K=1, KT)                -
C                                             -

```

CW	KT*MULT=NUMBER OF WORDS	-
C		-
CD	ZDF(K) DENSITY FACTOR FOR THE K-TH FINE MESH	-
C		-
C	-----	-
CEOF		-

FISSRC

The FISSRC file is a binary, code-dependent file containing the energy-group total fission source at each spatial fine-mesh point, i , that is,

$$\sum_g (v\Sigma_f)_{g,i} \phi_{g,i} ,$$

The FISSRC file is automatically produced by the Solver Module whenever fissions are present .

```
C*****-
C          DATE 02/21/95 -
C -
CF          FISSRC -
CE          CODE DEPENDENT FISSION SOURCE -
C -
C*****-
```

```
C-----
C -
C -
CN          THIS FILE PROVIDES THE FISSION SOURCE -
C -
C -
C-----
```

```
C-----
CR          FILE IDENTIFICATION -
C -
CL          HNAME, (HUSE(I) , I=1, 2) , IVERS -
C -
CW          1+3*MULT=NUMBER OF WORDS -
C -
CD          HNAME          HOLLERITH FILE NAME - FISSRC - (A6) -
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6) -
CD          IVERS          FILE VERSION NUMBER -
CD          MULT           DOUBLE PRECISION PARAMETER -
CD                   1- A6 WORD IS SINGLE WORD -
CD                   2- A6 WORD IS DOUBLE PRECISION WORD -
C -
C-----
```

```

C-----
CR          SPECIFICATIONS          (1D RECORD)          -
C
CL  NDIM,NGROUP ,NINTI ,NINTJ ,NINTK , ITER ,EFFK ,POWER ,NBLOK  -
C
CW  9 =NUMBER OF WORDS          -
C
CD  NDIM          NUMBER OF DIMENSIONS          -
CD  NGROUP        NUMBER OF ENERGY GROUPS      -
CD  NINTI         NUMBER OF FIRST DIMENSION FINE MESH INTERVALS -
CD  NINTJ         NUMBER OF SECOND DIMENSION FINE MESH INTERVALS -
CD  NINTK         NUMBER OF THIRD DIMENSION FINE MESH INTERVALS -
CD  ITER          OUTER ITERATION NUMBER AT WHICH FISSION WAS -
CD                WRITTEN                          -
CD  EFFK          EFFECTIVE MULTIPLICATION FACTOR -
CD  POWER        POWER IN WATTS TO WHICH FISSION IS NORMALIZED -
CD  NBLOK        SET TO 1                          -
C
C-----

C-----
CR          FISSION SOURCE          (2D RECORD)          -
C
CL  ((FISS(I,J) , I=1 ,NINTI) , J=1 ,NINTJ)          -
C
CW  NINTI*NINTJ*MULT=NUMBER OF WORDS          -
C
C      DO 1 K=1, NINTK          -
C      1 READ (N) *LIST AS ABOVE*          -
C
CD  FISS(I,J)          FISSION SOURCE WITHOUT VOLUME AT FINE MESH -
CD                    POINT (I,J) IN PLANE K, I.E., NUSIGF*FLUX -
C
C-----
CEOF

```

GEODST

This GEODST file is an extended version of the standard GEODST file. It is a binary, code-dependent file containing the geometry description. The standard GEODST file is a subset of this extended version.

```
C*****-
C   REVISED 11/30/76   EXTENDED 11/14/90   EXTENSION REVISED 12/11/03   -
C   -
CF      GEODST - IV   -
C   -
CE      GEOMETRY DESCRIPTION   -
C   -
C*****
```

```
C-----
CS   FILE STRUCTURE   -
CS   -
CS   RECORD TYPE           PRESENT IF   -
CS   =====           =====   -
CS   FILE IDENTIFICATION   ALWAYS   -
CS   FILE SPECIFICATIONS   ALWAYS   -
CS   ONE DIMENSIONAL COARSE MESH   IGOM.GT.0 AND IGOM.LE.3   -
CS   TWO DIMENSIONAL COARSE MESH   IGOM.GE.6 AND IGOM.LE.11   -
CS   THREE DIMENSIONAL COARSE MESH   IGOM.GE.12 AND IGOM.LE.18   -
CS   GEOMETRY DATA           IGOM.GT.0 OR NBS.GT.0   -
CS   BLOCK LEVELS             ILEVEL.GT.0   -
CS   REGION ASSIGNMENTS TO COARSE MESH   IGOM.GT.0 AND NRASS.EQ.0   -
CS   REGION ASSIGNMENTS TO FINE MESH   IGOM.GT.0 AND NRASS.EQ.1   -
C   -
C-----
```

```
C-----
CR   FILE IDENTIFICATION (0V RECORD)   -
C   -
CL   HNAME, (HUSE(I), I=1, 2), IVERS   -
C   -
CW   1+3*MULT   -
C   -
CD   HNAME           HOLLERITH FILE NAME - GEODST - (A6)   -
CD   HUSE           HOLLERITH USER IDENTIFICATION (A6)   -
CD   IVERS          FILE VERSION NUMBER   -
CD   MULT           DOUBLE PRECISION PARAMETER   -
CD   1- A6 WORD IS SINGLE WORD   -
CD   2- A6 WORD IS DOUBLE PRECISION WORD   -
C   -
C-----
```

```
C-----
CR   FILE SPECIFICATIONS (1D RECORD)   -
C   -
CL   IGOM, NZONE, NREG, NZCL, NCINTI, NCINTJ, NCINTK, NINTI, NINTJ, NINTK, IMB1, -
CL   IMB2, JMB1, JMB2, KMB1, KMB2, NBS, NBSC, NIBCS, NZWBB, NTRIAG, NRASS, NTHPT, -
CL   (NGOP(I), I=1, 3), ILEVEL   -
C   -
CW   27   -
C-----
```

```

C
CD   IGOM          GEOMETRY  0- POINT (FUNDAMENTAL MODE)           -
CD                                     1- SLAB                       -
CD                                     2- CYLINDER                      -
CD                                     3- SPHERE                        -
CD                                     6- X-Y                          -
CD                                     7- R-Z                          -
CD                                     8- THETA-R                       -
CD                                     9- UNIFORM TRIANGULAR             -
CD                                     10- HEXAGONAL (1 MESH POINT IN EACH -
CD                                           HEXAGONAL ELEMENT)           -
CD                                     11- R-THETA                       -
CD                                     12- R-THETA-Z                    -
CD                                     13- R-THETA-ALPHA                 -
CD                                     14- X-Y-Z                          -
CD                                     15- THETA-R-Z                      -
CD                                     16- THETA-R-ALPHA                 -
CD                                     17- UNIFORM TRIANGULAR-Z          -
CD                                     18- HEXAGON-Z (MESH POINTS AS IN 10 -
CD                                           ABOVE)                       -
CD                                     106- X-Y GENERALIZED                -
CD                                     107- R-Z GENERALIZED                -
C
CD   NZONE         NUMBER OF ZONES (EACH HOMOGENEOUS IN NEUTRONICS -
CD                                     PROBLEM - A ZONE CONTAINS ONE OR MORE REGIONS) -
CD   NREG          NUMBER OF REGIONS                               -
CD   NCZCL         NUMBER OF ZONE CLASSIFICATIONS (EDIT PURPOSES) -
CD   NCINTI        NUMBER OF FIRST DIMENSION COARSE MESH INTERVALS -
CD   NCINTJ        NUMBER OF SECOND DIMENSION COARSE MESH -
CD                                     INTERVALS. NCINTJ.EQ.1 FOR ONE DIMENSIONAL -
CD                                     CASE.                               -
CD   NCINTK        NUMBER OF THIRD DIMENSION COARSE MESH INTERVALS -
CD                                     NCINTK.EQ.1 FOR ONE AND TWO DIMENSIONAL -
CD                                     CASES.                               -
CD   NINTI         NUMBER OF FIRST DIMENSION FINE MESH INTERVALS   -
CD   NINTJ         NUMBER OF SECOND DIMENSION FINE MESH INTERVALS -
CD                                     NINTJ.EQ.1 FOR ONE DIMENSIONAL CASE.   -
CD   NINTK         NUMBER OF THIRD DIMENSION FINE MESH INTERVALS -
CD                                     NINTK.EQ.1 FOR ONE AND TWO DIMENSION CASES. -
CD   IMB1          FIRST BOUNDARY ON FIRST DIMENSION               -
CD                                     0 - ZERO FLUX (DIFFUSION)           -
CD                                     1 - REFLECTED                       -
CD                                     2 - EXTRAPOLATED (DIFFUSION - DEL PHI/PHI -
CD                                           = -C/D WHERE C IS GIVEN AS BNDC BELOW -
CD                                           AND D IS THE GROUP DIFFUSION CONSTANT, -
CD                                           TRANSPORT - NO RETURN).       -
CD                                     3 - REPEATING (PERIODIC) WITH OPPOSITE FACE -
CD                                     4 - REPEATING (PERIODIC) WITH NEXT ADJACENT -
CD                                           FACE.                           -
CD                                     5 - INVERTED REPEATING ALONG THIS FACE. -
CD                                           (180 DEGREE ROTATION)         -
CD                                     6 - ISOTROPIC RETURN (TRANSPORT) -
C
CC   NOTE FOR REPEATING CONDITIONS (3,4,5) - LET I1 DENOTE FIRST-
CC   BOUNDARY ON FIRST DIMENSION, I2 THE SECOND BOUNDARY ON THE -
CC   FIRST DIMENSION, J1 THE FIRST BOUNDARY ON THE SECOND -
CC   DIMENSION, ETC. THEN THESE REPEATING BOUNDARY CONDITIONS -
CC   ONLY APPLY TO BOUNDARIES I1,I2,J1, AND J2. GOING IN ORDER -
CC   OF I1,J1,I2,J2, THE FIRST BOUNDARY WHICH IS INVOLVED -
CC   CARRIES THE DESIGNATOR DEFINING THE REPEATING CONDITION. -
C
C
CD   IMB2          LAST BOUNDARY ON FIRST DIMENSION               -
CD   JMB1          FIRST BOUNDARY ON SECOND DIMENSION             -
CD   JMB2          LAST BOUNDARY ON SECOND DIMENSION             -

```

CD	KMB1	FIRST BOUNDARY ON THIRD DIMENSION	-
CD	KMB2	LAST BOUNDARY ON THIRD DIMENSION	-
CD	NBS	NUMBER OF BUCKLING SPECIFICATIONS	-
CD		0 - NONE	-
CD		1 - SINGLE VALUE APPLIES EVERYWHERE	-
CD		.EQ.NZONE- ZONE DEPENDENT	-
CD		M*NZONE - DATA IS GIVEN OVER ALL ZONES FOR	-
CD		THE FIRST ENERGY GROUP, THEN FOR THE	-
CD		NEXT GROUP, TO END OF LIST. IF	-
CD		M.LT.NGROUP THEN THE M-TH GROUP DATA	-
CD		APPLIES TO ALL ADDITIONAL GROUPS.	-
CD		(2.LE.M.LE.NGROUP)	-
CD	NBCS	NUMBER OF CONSTANTS FOR EXTERNAL BOUNDARIES	-
CD		0 - NONE	-
CD		1 - SINGLE VALUE USED EVERYWHERE	-
CD		6 - INDIVIDUAL VALUE GIVEN FOR EACH	-
CD		EXTERNAL BOUNDARY. THE ORDERING OF THE	-
CD		VALUES IS THE SAME AS THE ORDERING OF	-
CD		THE BOUNDARY CONDITIONS.	-
CD		6*M - SIX VALUES GIVEN FOR FIRST ENERGY	-
CD		GROUP (ORDERED AS DESCRIBED ABOVE),	-
CD		THEN 6 FOR THE NEXT GROUP, TO END OF	-
CD		LIST. (2.LE.M.LE.NGROUP).	-
CD		IF M.LT.NGROUP THEN THE M-TH GROUP DATA	-
CD		APPLIES TO ALL REMAINING GROUPS.	-
CD	NIBCS	NUMBER OF CONSTANTS FOR INTERNAL BOUNDARIES	-
CD		0 - NONE	-
CD		1 - SINGLE VALUE USED EVERYWHERE	-
CD		.GT.1 - VALUES ARE GIVEN BY ENERGY GROUP	-
CD		WITH NON-BLACK CONDITION INDICATED BY	-
CD		ZERO ENTRY - LAST VALUE APPLIES TO	-
CD		ADDITIONAL GROUPS	-
CD	NZWBB	NUMBER OF ZONES WHICH ARE BLACK ABSORBERS	-
CD	NTRIAG	TRIANGULAR/HEXAGONAL GEOMETRY OPTION	-
CD		0 - REGION OF SOLUTION IS A RHOMBUS IN	-
CD		WHICH THE 1ST AND 2ND DIMENSION AXES	-
CD		INTERSECT AT AN ANGLE OF 120 DEGREES.	-
CD		1 - REGION OF SOLUTION IS A RHOMBUS IN	-
CD		WHICH THE 1ST AND 2ND DIMENSION AXES	-
CD		INTERSECT AT AN ANGLE OF 60 DEGREES.	-
CD		2 - REGION OF SOLUTION IS A RECTANGLE. THE	-
CD		BOUNDARIES I1 AND I2 BISECT MESH	-
CD		TRIANGLES. SEE NTHPT BELOW.	-
CD		(IGOM=9,17 ONLY)	-
CD		3 - REGION OF SOLUTION IS AN EQUILATERAL,	-
CD		60 DEGREE TRIANGLE. (IGOM=9,17 ONLY)	-
CD		4 - REGION OF SOLUTION IS A 30-60 DEGREE	-
CD		RIGHT TRIANGLE IN WHICH THE 1ST AND 2ND	-
CD		DIMENSION AXES INTERSECT AT THE 30	-
CD		DEGREE ANGLE. (IGOM=9,17 ONLY)	-
CD		5 - REGION OF SOLUTION IS A RHOMBUS IN	-
CD		WHICH THE 1ST AND 2ND DIMENSION AXES	-
CD		INTERSECT AT AN ANGLE OF 30 DEGREES.	-
CD		(IGOM=9,17 ONLY)	-
CD	NRASS	REGION ASSIGNMENTS	-
CD		0- TO COARSE MESH	-
CD		1- TO FINE MESH	-
CD		2- TO SUBMESH	-
CD	NTHPT	ORIENTATION OF FIRST FINE MESH INTERVAL IN	-
CD		TRIANGULAR GEOMETRIES. NTRIAG=2 ONLY.	-
CD		1- TRIANGLE(1,1) POINTS AWAY FROM FIRST	-
CD		DIMENSION AXIS, I.E., NO INTERNAL MESH	-
CD		LINE INTERSECTS THE ORIGIN.	-
CD		2- TRIANGLE(1,1) POINTS TOWARD THE FIRST	-
CD		DIMENSION AXIS, I.E., AN INTERNAL MESH	-

```

CD          LINE INTERSECTS THE ORIGIN. -
CD  NGOP    RESERVED -
C   ILEVEL  PRESENCE OF BLOCK LEVELS -
C-----

C-----
CR          ONE DIMENSIONAL COARSE MESH INTERVAL BOUNDARIES AND FINE -
CR          MESH INTERVALS (2D RECORD) -
C - - - - - -
CC          PRESENT IF IGOM.GT.0 AND IGOM.LE.3 -
C - - - - - -
CL          (XMESH(I), I=1,NCBNDI), (IFINTS(I), I=1,NCINTI) -
C - - - - - -
CW          NCBNDI*MULT+NCINTI -
C - - - - - -
CD  XMESH   COARSE MESH BOUNDARIES, FIRST DIMENSION -
CD  IFINTS  NUMBER OF EQUALLY SPACED FINE MESH INTERVALS -
CD          PER COARSE MESH INTERVAL, FIRST DIMENSION. -
CD  NCBNDI  NCINTI+1, NUMBER OF FIRST DIMENSION COARSE MESH -
CD          BOUNDARIES -
C - - - - - -
CC          UNITS ARE CM FOR LINEAR DIMENSIONS AND RADIANS FOR ANGULAR -
CC          DIMENSIONS -
C - - - - - -
C-----

C-----
CR          TWO DIMENSIONAL COARSE MESH INTERVAL BOUNDARIES AND FINE -
CR          MESH INTERVALS (3D RECORD) -
C - - - - - -
CC          PRESENT IF IGOM.GE.6 AND IGOM.LE.11 -
C - - - - - -
CL          (XMESH(I), I=1,NCBNDI), (YMESH(J), J=1,NCBNDJ), -
CL  1 (IFINTS(I), I=1,NCINTI), (JFINTS(J), J=1,NCINTJ) -
C - - - - - -
CW          (NCBNDI+NCBNDJ)*MULT+NCINTI+NCINTJ -
C - - - - - -
CD  YMESH   COARSE MESH BOUNDARIES, SECOND DIMENSION -
CD  JFINTS  NUMBER OF EQUALLY SPACED FINE MESH INTERVALS -
CD          PER COARSE MESH INTERVAL, SECOND DIMENSION. -
CD  NCBNDJ  NCINTJ+1, NUMBER OF SECOND DIMENSION COARSE -
CD          MESH BOUNDARIES -
C - - - - - -
CC          FOR UNIFORM-TRIANGULAR-MESH GEOMETRY (IGOM = 9) THE -
CC          LENGTH (L) OF THE SIDE OF A MESH TRIANGLE MUST BE GIVEN -
CC          BY THE EXPRESSION -
CC           $L = 2. * (XMESH(2) - XMESH(1)) / IFINTS(1)$  . -
CC          FOR UNIFORM-HEXAGONAL-MESH GEOMETRY (IGOM = 10) THE -
CC          FLAT-TO-FLAT DISTANCE (FTF) ACROSS A MESH HEXAGON MUST -
CC          BE GIVEN BY THE EXPRESSION -
CC           $FTF = (XMESH(2) - XMESH(1)) / IFINTS(1)$  -
C - - - - - -
C-----

C-----
CR          THREE DIMENSIONAL COARSE MESH INTERVAL BOUNDARIES AND FINE -
CR          MESH INTERVALS (4D RECORD) -
C - - - - - -
CC          PRESENT IF IGOM.GE.12 .AND. IGOM.LT.100 -
C - - - - - -
CL          (XMESH(I), I=1,NCBNDI), (YMESH(J), J=1,NCBNDJ), -
CL  1 (ZMESH(K), K=1,NCBNDK), (IFINTS(I), I=1,NCINTI), -
CL  2 (JFINTS(J), J=1,NCINTJ), (KFINTS(K), K=1,NCINTK) -

```

```

C
CW      (NCBNDI+NCBNDJ+NCBNDK) *MULT+NCINTI+NCINTJ+NCINTK
C
CD      ZMESH          COARSE MESH BOUNDARIES, THIRD DIMENSION
CD      KFINTS         NUMBER OF EQUALLY SPACED FINE MESH INTERVALS
CD                        PER COARSE MESH INTERVAL, THIRD DIMENSION.
CD      NCBNDK         NCINTK+1, NUMBER OF THIRD DIMENSION COARSE MESH
CD                        BOUNDARIES
C
CC      FOR UNIFORM-TRIANGULAR-MESH GEOMETRY (IGOM = 17) THE
CC      LENGTH (L) OF THE SIDE OF A MESH TRIANGLE MUST BE GIVEN
CC      BY THE EXPRESSION
CC      L = 2. * (XMESH(2) - XMESH(1)) / IFINTS(1) .
CC      FOR UNIFORM-HEXAGONAL-MESH GEOMETRY (IGOM = 18) THE
CC      FLAT-TO-FLAT DISTANCE (FTF) ACROSS A MESH HEXAGON MUST
CC      BE GIVEN BY THE EXPRESSION
CC      FTF = (XMESH(2) - XMESH(1)) / IFINTS(1)
C
C-----

C-----
CR      GEOMETRY DATA (5D RECORD)
C
CC      PRESENT IF IGOM.GT.0 OR NBS.GT.0
C
CL      (VOLR(N), N=1, NREG), (BSQ(N), N=1, NBS), (BNDC(N), N=1, NBCS),
CL      (BNCI(N), N=1, NIBCS), ((NZHBB(N), N=1, NZWBB), (NZC(N), N=1, NZONE),
CL      (NZNR(N), N=1, NREG)
C
CW      2*NREG+NBS+NBCS+NIBCS+NZWBB+NZONE
C
CD      VOLR           REGION VOLUMES (CC)
CD      BSQ            BUCKLING (B**2) VALUES (CM**-2)
CD      BNDC           BOUNDARY CONSTANTS (DEL PHI/PHI = -C/D)
CD      BNCI           INTERNAL BLACK BOUNDARY CONSTANTS
CD      NZHBB          ZONE NUMBERS WITH BLACK ABSORBER CONDITIONS
CD      NZC            ZONE CLASSIFICATIONS
CD      NZNR           ZONE NUMBER ASSIGNED TO EACH REGION
C
C-----

C-----
CR      BLOCK LEVELS (5.5D RECORD)
C
CC      PRESENT IF ILEVEL.GT.0
C
CL      (LEVEL(I, J, K), I=NCJNTI, J=1, NCINTJ)
C
CW      NCINTI*NCINTK
C
CS      DO 1 K=1, NCINTK
CS      1 READ(N) *LIST AS ABOVE*
C
C-----

C-----
CR      REGION ASSIGNMENTS TO COARSE MESH INTERVALS (6D RECORD)
C
CC      PRESENT IF IGOM.GT.0 AND NRASS.EQ.0
C
CL      ((MR(I, J), I=1, NCINTI), J=1, NCINTJ) ----NOTE STRUCTURE BELOW----
C

```

```

CW      NCINTI*NCINTJ      -
C      -
CS      DO 1 K=1,NCINTK    -
CS 1    READ(N) *LIST AS ABOVE*  -
C      -
CD      MR                REGION NUMBERS ASSIGNED TO COARSE MESH  -
CD                        INTERVALS      -
C      -
C-----

```

```

C-----
CR      REGION ASSIGNMENTS TO FINE MESH INTERVALS (7D RECORD)  -
C      -
CC      PRESENT IF IGOM.GT.0 AND NRASS.EQ.1  -
C      -
CL      ((MR(I,J),I=1,NINTI),J=1,NINTJ) ----NOTE STRUCTURE BELOW----  -
C      -
CW      NINTI*NINTJ      -
C      -
CS      DO 1 K=1,NINTK    -
CS 1    READ(N) *LIST AS ABOVE*  -
C      -
CD      MR                REGION NUMBERS ASSIGNED TO FINE MESH  -
CD                        INTERVALS      -
C      -
C-----

```


LNK3DNT

The LNK3DNT file is a binary, code dependent file to enable the mixing of macroscopic cross sections on the fine mesh by a volume fraction method. This is used in PARTISN only in either X-Y or X-Y-Z symmetries to model more complicated geometrical bodies in those symmetries.

```

C*****-
C              REVISED 02/22/02              -
C                                             -
CF          LNK3DNT                          -
CE          FINE MESH MIXING FILE            -
C                                             -
C*****-

C-----
C                                             -
C                                             -
CN   THIS FILE PROVIDES THE FINE MESH MIXING PRESCRIPTIONS -
C                                             -
C                                             -
C-----

C-----
C                                             -
C                                             -
CD          NOTE THAT DOUBLE PRECISION VOLUME FRACTIONS -
CD          ARE GIVEN WHEN MULT=2            -
C                                             -
C                                             -
C-----

C-----
CS   FILE STRUCTURE                          -
CS                                     -
CS   RECORD TYPE                            PRESENT IF -
CS   =====                               ===== -
CS   FILE IDENTIFICATION                    ALWAYS      -
CS   FILE SPECIFICATIONS                    ALWAYS      -
CS   TWO DIMENSIONAL COARSE MESH AND FINE  IGOM.EQ.6    -
CS   MESH INTERVALS                        -
CS   THREE DIMENSIONAL COARSE MESH AND    IGOM.EQ.14   -
CS   FINE MESH INTERVALS                  -
CS   FINE MESH INTERVAL WIDTHS           IVERS. EQ.5  -
CS   MIXING ARRAYS                        IVERS. GE.4  -
CS   MIXING ARRAYS                        IVERS. EQ.3  -
C                                             -
C-----

C-----
CR          FILE IDENTIFICATION              -
C                                             -
CL   HNAME, (HUSE(I), I=1,2), IVERS        -
C                                             -

```

```

CW      1+3*MULT=NUMBER OF WORDS      -
C      -
CD      HNAME          HOLLERITH FILE NAME - LNK3DNT - (A6)      -
CD      HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)      -
CD      IVERS          FILE VERSION NUMBER                      -
CD      MULT           DOUBLE PRECISION PARAMETER              -
CD                        1- A6 WORD IS SINGLE WORD            -
CD                        2- A6 WORD IS DOUBLE PRECISION WORD   -
C      -
C-----

```

```

C-----
CR      FILE SPECIFICATIONS      -
C      -
CL      IGOM,NZONE,RESERVED,RESERVED,NCINTI,NCINTJ,NCINTK,NINTI,
CL      NINTJ,NINTK,13*RESERVED,NMXSP,2*RESERVED, ILEVEL      -
C      -
CW      27      -
C      -
CD      IGOM          GEOMETRY    6- X-Y      -
CD                        14- X-Y-Z      -
C      -
CD      NZONE        NUMBER OF ZONES (EACH HOMOGENEOUS IN NEUTRONICS
CD                        PROBLEM - A ZONE CONTAINS ONE OR MORE REGIONS)
CD      NCINTI       NUMBER OF FIRST DIMENSION COARSE MESH INTERVALS
CD      NCINTJ       NUMBER OF SECOND DIMENSION COARSE MESH
CD                        INTERVALS. NCINTJ.EQ.1 FOR ONE DIMENSIONAL
CD                        CASE.
CD      NCINTK       NUMBER OF THIRD DIMENSION COARSE MESH INTERVALS
CD                        NCINTK.EQ.1 FOR ONE AND TWO DIMENSIONAL
CD                        CASES.
CD      NINTI        NUMBER OF FIRST DIMENSION FINE MESH INTERVALS
CD      NINTJ        NUMBER OF SECOND DIMENSION FINE MESH INTERVALS
CD                        NINTJ.EQ.1 FOR ONE DIMENSIONAL CASE.
CD      NINTK        NUMBER OF THIRD DIMENSION FINE MESH INTERVALS
CD                        NINTK.EQ.1 FOR ONE AND TWO DIMENSION CASES.
CD      NMXSP        NUMBER OF MIXING INSTRUCTIONS
C      ILEVEL       PRESENCE OF BLOCK LEVELS      -
C-----

```

```

C-----
CR      TWO DIMENSIONAL COARSE MESH INTERVAL BOUNDARIES AND FINE
CR      MESH INTERVALS      -
C      -
CC      PRESENT IF IGOM.EQ.6      -
C      -
CL      (XMESH(I),I=1,NCBNDI),(YMESH(J),J=1,NCBNDJ),
CL      1(IFINTS(I),I=1,NCINTI),(JFINTS(J),J=1,NCINTJ)
C      -
CW      (NCBNDI+NCBNDJ)*MULT+NCINTI+NCINTJ
C      -
CD      XMESH        COARSE MESH BOUNDARIES, FIRST DIMENSION
CD      IFINTS       NUMBER OF EQUALLY SPACED FINE MESH INTERVALS
CD                        PER COARSE MESH INTERVAL, FIRST DIMENSION.
CD      NCBNDI       NCINTI+1, NUMBER OF FIRST DIMENSION COARSE MESH
CD                        BOUNDARIES
C      -
CD      YMESH        COARSE MESH BOUNDARIES, SECOND DIMENSION
CD      JFINTS       NUMBER OF EQUALLY SPACED FINE MESH INTERVALS
CD                        PER COARSE MESH INTERVAL, SECOND DIMENSION.
CD      NCBNDJ       NCINTJ+1, NUMBER OF SECOND DIMENSION FINE
CD                        MESH BOUNDARIES
C      -
C-----

```

```

C-----
C-----
CR          THREE DIMENSIONAL COARSE MESH INTERVAL BOUNDARIES AND FINE -
CR          MESH INTERVALS                                         -
C                                                    -
CC          PRESENT IF IGOM.EQ.14                                   -
C                                                    -
CL          (XMESH (I) , I=1 ,NCBNDI) , (YMESH (J) , J=1 ,NCBNDJ) , -
CL          1 (ZMESH (K) , K=1 ,NCBNDK) , (IFINTS (I) , I=1 ,NCINTI) , -
CL          2 (JFINTS (J) , J=1 ,NCINTJ) , (KFINTS (K) , K=1 ,NCINTK) -
C                                                    -
CW          (NCBNDI+NCBNDJ+NCBNDK) *MULT+NCINTI+NCINTJ+NCINTK      -
C                                                    -
CD          ZMESH          COARSE MESH BOUNDARIES, THIRD DIMENSION -
CD          KFINTS         NUMBER OF EQUALLY SPACED FINE MESH INTERVALS -
CD                               PER COARSE MESH INTERVAL, THIRD DIMENSION. -
CD          NCBNDK         NCINTK+1, NUMBER OF THIRD DIMENSION FINE MESH -
CD                               BOUNDARIES                             -
C                                                    -
C-----

C-----
CR          FINE MESH INTERVAL WIDTHS                               -
C                                                    -
CC          PRESENT IF IVERS.EQ.5                                   -
C                                                    -
CW          NINTI                                                  -
CW          NINTJ                                                  -
CW          NINTK                                                  -
C                                                    -
CL          XH (I) , I=1 , NINTI                                     -
CL          YH (J) , J=1 , NINTJ                                     -
CL          ZH (K) , K=1 , NINTK                                     -
C                                                    -
CD          XH FINE MESH INTERVAL WIDTHS, FIRST DIM                -
CD          YH FINE MESH INTERVAL WIDTHS, SECOND DIM              -
CD          ZH FINE MESH INTERVAL WIDTHS, THIRD DIM               -
C                                                    -
C-----

C-----
CR          MIXING ARRAYS                                           -
C                                                    -
CC          PRESENT IF IVERS.GE.4                                   -
C                                                    -
CL          (IDC (I , J , K , N) , I=1 ,NINTI , J=1 ,NINTK , K=1 ,NINTK , N=1 , NMXSP) -
C                                                    -
CW          NINTI*NINTJ*NINTK*NMXSP                                -
C                                                    -
CD          IDC MACROSCOPIC MATERIAL NUMBER                        -
C                                                    -
CL          (DEN (I , J , K , N) , I=1 , NINTI , J=1 , NINTJ , K=1 , NINTK , N=1 , NMXSP) -
C                                                    -
CW          NINTI*NINTJ*NINTK*NMXSP*MULT                          -
C                                                    -
CD          DEN MATERIAL DENSITY                                   -
C                                                    -
C-----

```

```
C-----
CR          MIXING ARRAYS                                -
C                                                  -
CC    PRESENT IF IVERS.EQ.3                            -
C                                                  -
CL          (IDC(I), I=1, NMXSP)                        -
C                                                  -
CW    NMXSP=NUMBER OF WORDS                            -
C                                                  -
CL          (DEN(I), I=1, NMXSP)                        -
C                                                  -
CW    NMXSP*MULT=NUMBER OF WORDS                       -
C                                                  -
CL          (IPT(I), I=1, NMXSP)                        -
C                                                  -
CW    NMXSP=NUMBER OF WORDS                            -
C                                                  -
CD    IDC(I)          MACROSCOPIC MATERIAL NUMBER AT I -
CD    DEN(I)          VOLUME FRACTION AT I             -
CD    IPT(I)          FINE MESH CELL NUMBER AT I      -
C                                                  -
C-----
CEOF
```

MACRXS

The MACRXS code-dependent interface file is the working cross-section file for the Solver Module. On the MACRXS file are the material macroscopic cross sections arranged in energy-group order. The contents of this file are described below:

```

C*****
C          DATE 12/10/03
C
C          MACRXS
CE          CODE DEPENDENT MACROSCOPIC MULTIGROUP CROSS SECTION FILE
CE          FOR USE IN ONEDANT SOLVER MODULE
C
C
C*****
C
CN          THIS FILE PROVIDES A BASIC BROAD GROUP
CN          LIBRARY, ORDERED BY GROUP
C
C          ORDER OF GROUPS IS ACCORDING TO DECREASING ENERGY
C
C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE                PRESENT IF
CS          =====
CS          FILE IDENTIFICATION        ALWAYS
CS          FILE CONTROL                ALWAYS
CS          FILE DATA                 ALWAYS
CS
CS          ***** (REPEAT FOR ALL GROUPS)
CS          *          PRINCIPAL CROSS SECTIONS        ALWAYS
CS          *          SCATTERING CONTROL DATA        NORD.NE.0
CS          *          SCATTERING MATRIX              NORD.NE.0
CS          *****
CS
CS          FISSION WEIGHTING SPECTRUM        CHIMIX.NE.0
C
C-----
C
C-----
CR          FILE IDENTIFICATION
C
CL          HNAME, (HUSE(I), I=1, 2), IVERS
C
CW          1+3*MULT=NUMBER OF WORDS
C
CD          HNAME          HOLLERITH FILE NAME - MACRXS - (A6)
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)
CD          IVERS          FILE VERSION NUMBER
CD          MULT           DOUBLE PRECISION PARAMETER
CD                          1- A6 WORD IS SINGLE WORD
CD                          2- A6 WORD IS DOUBLE PRECISION WORD
C
C-----
C-----
CR          FILE CONTROL
C
CL          NGROUP, NMAT, NORD, NED, IDPF, LNG, MAXUP, MAXDN, NPRIN, I2LP1, CHIMIX

```

```

C
CW 10=NUMBER OF WORDS
C
CD NGROUP      NUMBER OF ENERGY GROUPS IN FILE
CD NMAT        NUMBER OF MATERIALS IN FILE
CD NORD        NUMBER OF LEGENDRE SCATTERING ORDERS
CD NED         NUMBER OF EXTRA EDIT CROSS SECTIONS (IN ADDITION
CD             TO THE BASIC PRINCIPAL CROSS SECTIONS)
CD IDPF        0/1 NO/YES CROSS SECTION DATA ARE DOUBLE PRECISION
CD LNG         NUMBER OF THE LAST NEUTRON GROUP(FOR COUPLED SETS)
CD MAXUP       MAXIMUM NUMBER OF UPSCATTER GROUPS
CD MAXDN       MAXIMUM NUMBER OF DOWNSCATTER GROUPS
CD NPRIN       NUMBER OF PRINCIPAL CROSS SECTIONS
CD I2LP1       0/1 = NO/YES 2L+1 TERM WAS INCLUDED IN LIBRARY
CD CHIMIX      0/1 = NO/YES ZONE MIXED CHI (NON-ZERO ONLY IF
CD             NDI IS USED AS CROSS SECTION LIBRARY
C
C-----
C
C-----
CR          FILE DATA
C
CL (HMAT(I), I=1, NMAT), (HED(J), J=1, NEDT), (VEL(N), N=1, NGROUP),
CL 1(EMAX(N), N=1, NGROUP), EMIN
C
CW (NMAT+NEDT+2*NGROUP+1)*MULT=NUMBER OF WORDS
C
CD HMAT(I)     HOLLERITH MATERIAL LABEL FOR MATERIAL I (A6)
CD HED(J)      HOLLERITH LABEL FOR J-TH CROSS SECTION POSITION(A6)
CD VEL(N)      MEAN NEUTRON VELOCITY IN GROUP N (CM/SEC)
CD EMAX(N)     MAXIMUM ENERGY BOUND OF GROUP N (EV)
CD EMIN        MINIMUM ENERGY BOUND OF SET (EV)
CD NEDT       NED+NPRIN
C
CN           THE FOUR BASIC PRINCIPAL CROSS SECTIONS
CN           ALWAYS PRESENT ARE:
CN
CN           HED(1) = 3HCHI
CN           HED(2) = 6HNUSIGF
CN           HED(3) = 5HTOTAL
CN           HED(4) = 3HABS
C
CN           ALSO PRESENT WHEN NPRIN=5 IS:
C
CN           HED(5) = 5HTRANS
C
C-----
C-----
CR          PRINCIPAL CROSS SECTIONS FOR GROUP N
C
CL ((C(I,J), I=1, NMAT), J=1, NEDT)
C
CW NMAT*NEDT*MULT=NUMBER OF WORDS
C
CD C(I,J)      PRINCIPAL CROSS SECTIONS
C
CN           BASIC PRINCIPAL CROSS SECTIONS ALWAYS PRESENT ARE:
CN
CN           J=1 FISSION SPECTRUM
CN           J=2 FISSION NU*FISSION CROSS SECTION
CN           J=3 TOTAL CROSS SECTION
CN           J=4 ABSORPTION CROSS SECTION
C
CN           ALSO PRESENT WHEN NPRIN=5 IS:

```

```

C
CN          J=5  TRANSPORT CROSS SECTION
C
C-----
C
C-----
CR          SCATTERING CONTROL BLOCK FOR GROUP N
C
CC          PRESENT IF NORD.GT.0
C
CL          ((NGPB(L,J),L=1,NORD),J=1,NMAT)
CL          ((IFSG(L,J),L=1,NORD),J=1,NMAT)
C
CW          2*NORD*NMAT=NUMBER OF WORDS
C
CD          NGPB(L,J)  NUMBER OF SOURCE GROUPS THAT CAN SCATTER INTO GROUP N-
CD          IFSG(L,J)  GROUP NUMBER OF THE FIRST SOURCE GROUP
CD          L          LEGENDRE ORDER NUMBER
CD          J          MATERIAL NUMBER
C
C-----
C-----
CR          SCATTERING SUB-BLOCK FOR GROUP N
C
CC          PRESENT IF NORD.GT.0
C
CL          (SCAT(I),I=1,NTAB)
C
CW          NTAB*MULT=NUMBER OF WORDS
C
CD          SCAT(I)    SCATTERING CROSS SECTION
C
CD          NTAB      TABLE LENGTH OF THE CROSS SECTIONS FOR SCATTERING
CD                   INTO GROUP N.  THIS IS FOR ALL MATERIALS AND ALL
CD                   LEGENDRE ORDERS,  THUS IT IS THE SUM OF NGPB(L,J)
CD                   OVER L FROM 1 TO NORD AND OVER J FROM 1 TO NMAT.
C
CN          THE SCATTERING CROSS SECTIONS ARE PACKED IN BANDS,
CN          ONE FOR EACH LEGENDRE ORDER AND MATERIAL.  EACH BAND-
CN          CONTAINS THE NGPB GROUPS WHICH SCATTER INTO GROUP
CN          N.  THE FIRST SOURCE GROUP NUMBER IS IFSG AND
CN          THE LAST IS IFSG-NGPB+1.  THE NORD BANDS FOR THE
CN          FIRST MATERIAL APPEAR FIRST (P0, P1, ...) FOLLOWED-
CN          BY THE NORD BANDS FOR THE SECOND, ETC.
C
CN          HIGHER LEGENDRE ORDER SCATTERING CROSS SECTIONS
CN          INCLUDE A 2*L+1 FACTOR WHERE L IS THE LEGENDRE
CN          ORDER.
C
C-----
C-----
CR          FISSION WEIGHTING SPECTRUM
C
CC          PRESENT IF CHIMIX.NE.0
C
CL          (FISS_WGT(I),I=1,NMAT)
C
CW          NMAT=NUMBER OF WORDS
C
CD          FISS_WGT(I)  FISSION WEIGHTING SPECTRUM FOR MATERIAL I
C
C-----
C-----
CEOF

```


RAFLXM

The RAFLXM file is a binary, code-dependent file containing the angular fluxes for the flux at each fine-mesh boundary. It differs from the standard angular flux file RAFLUX only in that the fluxes are in different order. In RAFLXM, the fluxes are in calculational order.

```

C*****-
C                                     DATE 03/03/95 -
C                                     -
CF          RAFLXM -
CE          CODE DEPENDENT COMPUTATIONAL-ORDERED ANGULAR FLUX AT CELL -
CE          EDGES FOR PARTISN CODE -
C                                     -
C*****-

C-----
C                                     -
C                                     -
CN          THIS FILE PROVIDES THE EDGE ANGULAR FLUX AS ORDERED -
CN          BY THE CODE -
C                                     -
C                                     -
C-----

C-----
C                                     -
CD          ORDER OF GROUPS IS ACCORDING TO DECREASING -
CD          ENERGY. NOTE THAT DOUBLE PRECISION FLUXES ARE -
CD          GIVEN WHEN MULT=2 -
C                                     -
C-----

C-----
CS          FILE STRUCTURE -
CS          RECORD TYPE                PRESENT IF -
CS          ===== -
CS          FILE IDENTIFICATION        ALWAYS -
CS          FILE CONTROL                ALWAYS -
CS          -
CS          ***** (REPEAT FOR ALL GROUPS) -
CS          *          (GROUP 1 IS FIRST) -
CS          *          ***** (REPEAT FOR ALL FRONT-GOING DIRECTIONS) -
CS          *          *          COMPUTATIONAL COSINES                ALWAYS -
CS          *          *          CELL BACK-EDGE ANGULAR FLUX FOR BACK PLANE -
CS          *          ***** -
CS          *          -
CS          *          ***** (REPEAT FOR ALL Z-INTERVALS) -
CS          *          (INTERVAL NINTK IS FIRST) -
CS          *          *          ***** (REPEAT FOR ALL FRONT-GOING DIRECTIONS) -
CS          *          *          *          COMPUTATIONAL COSINES                ALWAYS -
CS          *          *          *          HORIZONTAL-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          *          VERTICAL-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          *          CELL FRONT-EDGE ANGULAR FLUX        ALWAYS -
CS          *          *          ***** -

```

```

CS * * ***** -
CS * -
CS * ***** (REPEAT FOR ALL BACK-GOING DIRECTIONS) -
CS * * COMPUTATIONAL COSINES ALWAYS -
CS * * CELL FRONT-EDGE ANGULAR FLUX FOR FRONT PLANE -
CS * ***** -
CS * -
CS * ***** (REPEAT FOR ALL Z-INTERVALS) -
CS * (INTERVAL 1 IS FIRST) -
CS * * ***** (REPEAT FOR ALL BACK-GOING DIRECTIONS) -
CS * * * COMPUTATIONAL COSINES ALWAYS -
CS * * * HORIZONTAL-EDGE ANGULAR FLUX ALWAYS -
CS * * * VERTICAL-EDGE ANGULAR FLUX ALWAYS -
CS * * * CELL BACK-EDGE ANGULAR FLUX ALWAYS -
CS * * ***** -
CS * * ***** -
CS ***** -
C -
C-----

```

```

C-----
CR FILE IDENTIFICATION -
C -
CL HNAME, (HUSE(I), I=1, 2), IVERS -
C -
CW 1+3*MULT=NUMBER OF WORDS -
C -
CD HNAME HOLLERITH FILE NAME - RAFLXM - (A6) -
CD HUSE(I) HOLLERITH USER IDENTIFICATION (A6) -
CD IVERS FILE VERSION NUMBER -
CD MULT DOUBLE PRECISION PARAMETER -
CD 1- A6 WORD IS SINGLE WORD -
CD 2- A6 WORD IS DOUBLE PRECISION WORD -
C -
C-----

```

```

C-----
CR SPECIFICATIONS (1D RECORD) -
C -
CL NDIM, NGROUP, NINTI, NINTJ, NINTK, EFFK, POWER -
C -
CW 8 =NUMBER OF WORDS -
C -
CD NDIM NUMBER OF DIMENSIONS -
CD NGROUP NUMBER OF ENERGY GROUPS -
CD NINTI NUMBER OF FIRST DIMENSION FINE MESH INTERVALS -
CD NINTJ NUMBER OF SECOND DIMENSION FINE MESH INTERVALS -
CD NINTK NUMBER OF THIRD DIMENSION FINE MESH INTERVALS. -
CD NINTK.EQ.1 IF NDIM.LE.2 -
CD EFFK EFFECTIVE MULTIPLICATION FACTOR -
CD POWER POWER IN WATTS TO WHICH FLUX IS NORMALIZED -
C -
C-----

```

```

C-----
CR COMPUTATIONAL COSINES -
C -
CL XMU, XETA, XI, WEIGHT, M, K -
C -
CW 6*MULT -
C -
CD XMU MU COSINE -

```

```

CD   XETA           ETA COSINE           -
CD   XI             XI COSINE           -
CD   WEIGHT        WEIGHT                -
CD   M             ANGLE INDEX IN THE OCTANT -
CD   K             Z-PLANE NUMBER        -
C                                     -
C-----

```

```

C-----
CR           CELL FRONT-EDGE ANGULAR FLUX -
C                                     -
CL   ((FBEDGE(I,J),I=NINTI),J=1,NINTJ) -
C                                     -
CW   NINTI*NINTJ*MULT=NUMBER OF WORDS -
C                                     -
CD   FBEDGE(I,J)   CELL FRONT-EDGE ANGULAR FLUX -
C                                     -
C-----

```

```

C-----
CR           HORIZONTAL-EDGE ANGULAR FLUX -
C                                     -
CL   ((HEDGE(I,J),I=1,NBDRYI),J=1,NINTJ) -
C                                     -
CD   HEDGE(I,J)   HORIZONTAL-EDGE-BOUNDARY ANGULAR FLUX -
C                                     -
CW   NBDRYI*NINTJ*MULT=NUMBER OF WORDS -
C                                     -
CD   NBDRYI       NINTI+1 (NUMBER OF FIRST DIMENSION FINE MESH -
CD                   BOUNDARIES) -
C                                     -
C-----

```

```

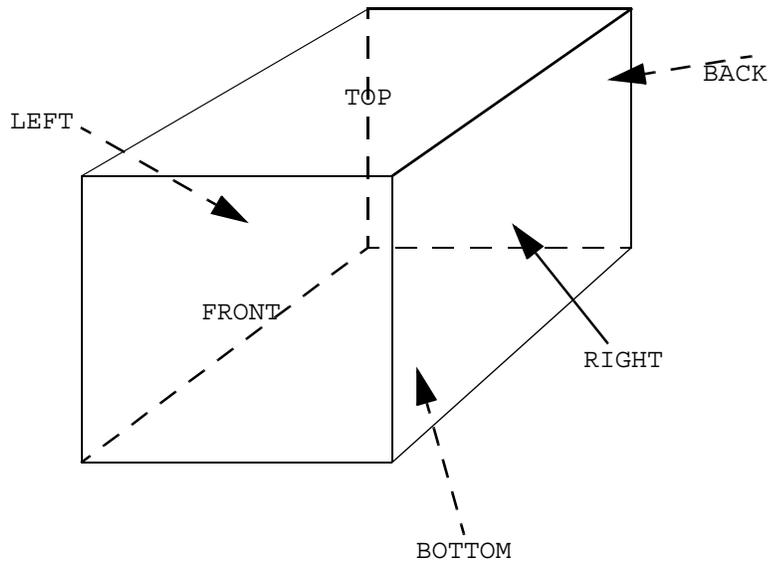
C-----
CR           VERTICAL-EDGE ANGULAR FLUX -
C                                     -
CL   ((VEDGE(I,J),I=1,NINTI),J=1,NBDRYJ) -
C                                     -
CD   VEDGE(I,J)   VERTICAL-EDGE-BOUNDARY ANGULAR FLUX -
C                                     -
CW   NINTI*NBDYJ*MULT=NUMBER OF WORDS -
C                                     -
CD   NBDRYJ       NINTJ+1 (NUMBER OF SECOND DIMENSION FINE MESH -
CD                   BOUNDARIES) -
C                                     -
C-----

```

```

C-----
CR           CELL BACK-EDGE ANGULAR FLUX -
C                                     -
CL   ((FBEDGE(I,J),I=NINTI),J=1,NINTJ) -
C                                     -
CW   NINTI*NINTJ*MULT=NUMBER OF WORDS -
C                                     -
CD   FBEDGE(I,J)   CELL BACK-EDGE ANGULAR FLUX -
C                                     -
C-----

```



CEOF

RMFLUX

The RMFLUX code-dependent file contains, in binary form, the spherical harmonics regular angular flux moments for all spatial fine mesh points and all energy groups. It is optionally produced by the Solver Module.

```

C*****
C          DATE 04/01/85
C
CF          RMFLUX-IV
CE          REGULAR FLUX MOMENTS
C
C*****
C
CN          ORDER OF GROUPS IS ACCORDING TO DECREASING ENERGY
C
C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE                PRESENT IF
CS          =====
CS          FILE IDENTIFICATION        ALWAYS
CS          SPECIFICATIONS              ALWAYS
CS
CS          ***** (REPEAT FOR ALL GROUPS)
CS          *          REGULAR MOMENTS FLUXES          ALWAYS
CS          *****
C
C-----
C
CR          FILE IDENTIFICATION
C
CL          HNAME, (HUSE(I), I=1,2), IVERS
C
CW          1+3*MULT=NUMBER OF WORDS
C
CD          HNAME          HOLLERITH FILE NAME - RMFLUX - (A6)
CD          HNAME          HOLLERITH FILE NAME -          - (A6)
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6)
CD          IVERS          FILE VERSION NUMBER
CD          MULT           DOUBLE PRECISION PARAMETER
CD                        1- A6 WORD IS SINGLE WORD
CD                        2- A6 WORD IS DOUBLE PRECISION WORD
C
C-----
C
CR          SPECIFICATIONS          (1D RECORD)
C
CL          NDIM, NGROUP, NINTI, NINTJ, NINTK, NORD, EFFK, POWER, OITNO
C
CW          9=NUMBER OF WORDS
C
CD          NDIM           NUMBER OF DIMENSIONS
CD          NGROUP         NUMBER OF GROUPS
CD          NINTI          NUMBER OF FIRST DIMENSION INTERVALS
CD          NINTJ          NUMBER OF SECOND DIMENSION INTERVALS
CD                        NINTJ.EQ.1 IF NDIM.EQ.1
CD          NINTK          NUMBER OF THIRD DIMENSION INTERVALS

```

```

CD          NINTK.EQ.1 IF NDIM.LE.2          -
CD  NORD    NUMBER OF LEGENDRE MOMENTS      -
CD  EFFK    EFFECTIVE MULTIPLICATION FACTOR -
CD  POWER   POWER IN WATTS TO WHICH FLUX IS -
CD  OITNO   OUTER ITERATION NUMBER         -
C-----
C
C-----
CR          REGULAR MOMENTS FLUXES ON MULTIDIMENSIONAL INTERVALS -
C          (2D RECORD)
C
CL  ((FLUX(M,I),M=1,NORD),I=1,NINTI)  ----NOTE STRUCTURE BELOW--- -
C
CW  NORD*NINTI=NUMBER OF WORDS
C
C  DO 1 K=1,NINTK
C  DO 1 J=1,NINTJ
C  1 READ (N) *LIST AS ABOVE*
C
CD  FLUX(M,I)    REGULAR FLUX MOMENTS ON FIRST DIMENSION -
CD              INTERVALS.
C
C-----
CEOF

```

RZMFLX

The RZMFLX file is a binary, code-dependent file containing the spherical harmonics regular angular flux moments averaged over each zone for each energy group. The zones over which the fluxes are averaged are the zones used in the Solver Module and not the Edit Zones optionally used in the Edit Module.

```

C*****
C                                DATE 01/28/95                                -
C                                                                           -
CF                                RZMFLX-IV                                -
CE                                REGULAR ZONE AVERAGED FLUX MOMENTS BY GROUP -
C                                                                           -
C*****
C                                                                           -
CN                                ORDER OF GROUPS IS ACCORDING TO DECREASING ENERGY -
C                                                                           -
C-----
CS                                FILE STRUCTURE                                -
CS                                                                           -
CS                                RECORD TYPE                                PRESENT IF                                -
CS                                =====                                =====                                -
CS                                FILE IDENTIFICATION                                ALWAYS                                -
CS                                SPECIFICATIONS                                ALWAYS                                -
CS                                                                           -
CS                                ***** (REPEAT FOR ALL GROUPS)                                -
CS                                *          ZONE AVERAGED FLUX MOMENTS                                ALWAYS                                -
CS                                *****                                -
C                                                                           -
C-----
C-----
CR                                FILE IDENTIFICATION                                -
C                                                                           -
CL                                HNAME, (HUSE(I), I=1, 2), IVERS                                -
C                                                                           -
CW                                1+3*MULT=NUMBER OF WORDS                                -
C                                                                           -
CD                                HNAME                                HOLLERITH FILE NAME - RZMFLX - (A6)                                -
CD                                HNAME                                HOLLERITH FILE NAME - (A6)                                -
CD                                HUSE(I)                                HOLLERITH USER IDENTIFICATION (A6)                                -
CD                                IVERS                                FILE VERSION NUMBER                                -
CD                                MULT                                DOUBLE PRECISION PARAMETER                                -
CD                                1- A6 WORD IS SINGLE WORD                                -
CD                                2- A6 WORD IS DOUBLE PRECISION WORD                                -
C                                                                           -
C-----
C-----
CR                                SPECIFICATIONS (1D RECORD)                                -
C                                                                           -
CL                                NDIM, NGROUP, NZONE, DUM, DUM, NORD, EFFK, POWER, OITNO                                -
C                                                                           -
CW                                9=NUMBER OF WORDS                                -
C                                                                           -
CD                                NDIM                                NUMBER OF DIMENSIONS                                -
CD                                NGROUP                                NUMBER OF GROUPS                                -
CD                                NZONE                                NUMBER OF GEOMETRIC ZONES                                -
CD                                DUM                                DUMMY, NOT USED                                -

```

```

CD   DUM           DUMMY, NOT USED           -
CD   NORD          NUMBER OF LEGENDRE MOMENTS -
CD   EFFK          EFFECTIVE MULTIPLICATION  -
CD   POWER         POWER IN WATTS TO WHICH   -
CD   OITNO         OUTER ITERATION NUMBER    -
C                                         -
C-----
C                                         -
C-----
CR           REGULAR FLUX MOMENTS AVERAGED OVER EACH ZONE -
C                                     (2D RECORD)          -
C                                         -
CL   (( FLUX (M, I) , M=1 , NORD) , I=1 , NZONE) -
C                                         -
CW   NORD*NZONE=NUMBER OF WORDS           -
C                                         -
C                                         -
CD   FLUX (M, I)   REGULAR FLUX MOMENT AVERAGES FOR EACH -
CD                                     ZONE .              -
C                                         -
C-----
CEOF                                         -

```

SNXEDT

The SNXEDT file is the working cross-section file for the Edit Module. On the SNXEDT file are the isotope microscopic cross sections arranged in energy-group order. Although the Edit Module will read any SNXEDT file constructed as described below, all SNXEDT files created by the PARTISN Input Module will have the parameter NORD set to zero so that scattering cross sections will not appear on the created SNXEDT file.

```

C*****-
C          DATE 05/12/83 -
C -
CF          SNXEDT -
CE          CODE DEPENDENT MICROSCOPIC MULTIGROUP CROSS SECTION FILE -
CE          FOR USE IN ONEDANT EDITS -
C -
C*****-
C -
C          THIS FILE PROVIDES A BASIC BROAD GROUP -
CN          LIBRARY, ORDERED BY GROUP -
C -
C -
C-----
CS          FILE STRUCTURE -
CS -
CS          RECORD TYPE                PRESENT IF -
CS          ===== -
CS          FILE IDENTIFICATION        ALWAYS -
CS          FILE CONTROL                ALWAYS -
CS          FILE DATA                 ALWAYS -
CS -
CS          ***** (REPEAT FOR ALL GROUPS) -
CS          *          (GROUP 1 IS FIRST) -
CS          *          PRINCIPAL CROSS SECTIONS        ALWAYS -
CS          *          SCATTERING CONTROL DATA        NORD.NE.0 -
CS          *          SCATTERING MATRIX              NORD.NE.0 -
CS          ***** -
C -
C-----
C -
C-----
CR          FILE IDENTIFICATION -
C -
CL          HNAME, (HUSE(I), I=1, 2), IVERS -
C -
CW          1+3*MULT=NUMBER OF WORDS -
C -
CD          HNAME          HOLLERITH FILE NAME - SNXEDT - (A6) -
CD          HUSE(I)        HOLLERITH USER IDENTIFICATION (A6) -
CD          IVERS          FILE VERSION NUMBER -
CD          MULT           DOUBLE PRECISION PARAMETER -
CD          1- A6 WORD IS SINGLE WORD -
CD          2- A6 WORD IS DOUBLE PRECISION WORD -
C -
C-----
C-----

```

```

CR          FILE CONTROL -
C          -
CL  NGROUP, NISO, NORD, NED, IDPF, LNG, MAXUP, MAXDN, NPRIN, I2LP1 -
C          -
CW  10=NUMBER OF WORDS -
C          -
CD  NGROUP      NUMBER OF ENERGY GROUPS IN FILE -
CD  NISO        NUMBER OF ISOTOPES IN FILE -
CD  NORD        NUMBER OF LEGENDRE SCATTERING ORDERS -
CD  NED         NUMBER OF EXTRA EDIT CROSS SECTIONS (IN ADDITION -
CD              TO THE BASIC PRINCIPAL CROSS SECTIONS) -
CD  IDPF        0/1 NO/YES CROSS SECTION DATA ARE DOUBLE PRECISION -
CD  LNG         NUMBER OF THE LAST NEUTRON GROUP (FOR COUPLED SETS) -
CD  MAXUP       MAXIMUM NUMBER OF UPSCATTER GROUPS -
CD  MAXDN       MAXIMUM NUMBER OF DOWNSCATTER GROUPS -
CD  NPRIN       NUMBER OF PRINCIPAL CROSS SECTIONS (4 FOR SNXEDT) -
CD  I2LP1       0/1 = NO/YES 2L+1 TERM WAS INCLUDED IN LIBRARY -
C          -
C-----
C          -
C          -
C-----
CR          FILE DATA -
C          -
CL  (HISO(I), I=1, NISO), (HED(J), J=1, NEDT), (VEL(N), N=1, NGROUP), -
CL  1 (EMAX(N), N=1, NGROUP), EMIN -
C          -
CW  (NISO+NEDT+2*NGROUP+1)*MULT=NUMBER OF WORDS -
C          -
CD  HISO(I)     HOLLERITH ISOTOPE LABEL FOR ISOTOPE (A6) -
CD  HED(J)      HOLLERITH LABEL FOR EDIT NUMBER J (A6) -
CD  VEL(N)      MEAN NEUTRON VELOCITY IN GROUP N (CM/SEC) -
CD  EMAX(N)     MAXIMUM ENERGY BOUND OF GROUP N (EV) -
CD  EMIN        MINIMUM ENERGY BOUND OF SET (EV) -
CD  NEDT        NED+NPRIN -
C          -
CN              THE FOUR BASIC PRINCIPAL CROSS SECTIONS -
CN              ALWAYS PRESENT ARE: -
CN              -
CN              HED(1) = 3HCHI -
CN              HED(2) = 6HNUSIGF -
CN              HED(3) = 5HTOTAL -
CN              HED(4) = 3HABS -
C          -
CN              ALSO PRESENT WHEN NPRIN=5 IS: -
C          -
CN              HED(5) = 5HTRANS -
C          -
C-----
C          -
CR          PRINCIPAL CROSS SECTIONS FOR GROUP N -
C          -
CL  ((C(I, J), I=1, NISO), J=1, NEDT) -
C          -
CW  NISO*NEDT*MULT=NUMBER OF WORDS -
C          -
CD  C(I, J)     PRINCIPAL CROSS SECTIONS -
C          -
CN              BASIC PRINCIPAL CROSS SECTIONS ALWAYS PRESENT ARE: -
CN              -
CN              J=1  FISSION SPECTRUM -
CN              J=2  FISSION NU*FISSION CROSS SECTION -
CN              J=3  TOTAL CROSS SECTION -
CN              J=4  ABSORPTION CROSS SECTION -
C          -

```

```

CN          ALSO PRESENT WHEN NPRIN=5 IS:          -
C                                                  -
CN          J=5  TRANSPORT CROSS SECTION          -
C                                                  -
C-----
C                                                  -
C-----
CR          SCATTERING CONTROL BLOCK FOR GROUP N  -
C                                                  -
CC          PRESENT IF NORD.GT.0                  -
C                                                  -
CL          ((NGPB(L,J),L=1,NORD),J=1,NISO)        -
CL          ((IFSG(L,J),L=1,NORD),J=1,NISO)        -
C                                                  -
CW          2*NORD*NISO=NUMBER OF WORDS           -
C                                                  -
CD          NGPB(L,J)  NUMBER OF SOURCE GROUPS THAT CAN SCATTER INTO GROUP N-
CD          IFSG(L,J)  GROUP NUMBER OF THE FIRST SOURCE GROUP  -
CD          L          LEGENDRE ORDER NUMBER        -
CD          J          ISOTOPE NUMBER               -
C                                                  -
C-----
C-----
CR          SCATTERING SUB-BLOCK FOR GROUP N      -
C                                                  -
CC          PRESENT IF NORD.GT.0                  -
C                                                  -
CL          (SCAT(I),I=1,NTAB)                    -
C                                                  -
CW          NTAB*MULT=NUMBER OF WORDS             -
C                                                  -
CD          SCAT(I)   SCATTERING CROSS SECTION    -
C                                                  -
CD          NTAB      TABLE LENGTH OF THE CROSS SECTIONS FOR SCATTERING -
CD                    INTO GROUP N.  THIS IS FOR ALL ISOTOPES AND ALL  -
CD                    LEGENDRE ORDERS,  THUS IT IS THE SUM OF NGPB(L,J)  -
CD                    OVER L FROM 1 TO NORD AND OVER J FROM 1 TO NISO.  -
C                                                  -
CN                    THE SCATTERING CROSS SECTIONS ARE PACKED IN BANDS, -
CN                    ONE FOR EACH LEGENDRE ORDER AND ISOTOPE.  EACH BAND -
CN                    CONTAINS THE NGPB GROUPS WHICH SCATTER INTO GROUP  -
CN                    N.  THE FIRST SOURCE GROUP NUMBER IS IFSG AND      -
CN                    THE LAST IS IFSG-NGPB+1.  THE NORD BANDS FOR THE  -
CN                    FIRST ISOTOPE APPEAR FIRST (P0, P1, ...) FOLLOWED  -
CN                    BY THE NORD BANDS FOR THE SECOND, ETC.            -
C                                                  -
CN                    HIGHER LEGENDRE ORDER SCATTERING CROSS SECTIONS  -
CN                    INCLUDE A 2*L+1 FACTOR WHERE L IS THE LEGENDRE    -
CN                    ORDER.                                             -
C-----
CEOF

```

SOLINP

The SOLINP code-dependent interface file contains information specific to the Solver Module, mainly the information from Block-V of the card-image input.

```

C*****
C                                DATE 01/28/95                                -
C                                                                           -
CF          SOLINP                                                         -
CE          CODE DEPENDENT FILE OF INFORMATION SPECIFIC TO THE             -
CE          SOLVER MODULE                                                 -
C                                                                           -
C                                                                           -
C*****
C-----
CS          FILE STRUCTURE                                               -
CS                                                                           -
CS          RECORD TYPE                                                    PRESENT IF                          -
CS          =====                                                    =====                          -
CS          FILE IDENTIFICATION                                           ALWAYS                          -
CS          TITLE CARD COUNT                                              ALWAYS                          -
CS          ***** (REPEAT FOR UP TO 10 CDS)                             -
CS          * TITLE CARD                                                  NHEAD.GT.0                      -
CS          *****                                                       -
CS          DIMENSION                                                      ALWAYS                          -
CS          RAW CONTROLS AND DIMENSIONS                                    ALWAYS                          -
CS          RAW FLOATING INPUT DATA                                       ALWAYS                          -
CS          DEFAULTED CONTROLS AND DIMENSIONS                              ALWAYS                          -
CS          DEFAULTED FLOATING INPUT DATA                                  ALWAYS                          -
CS          FINE MESH DENSITY FACTORS                                       IDENX=2                          -
CS          FINE MESH DENSITY VECTOR IN X                                  IDENX=1                          -
CS          FINE MESH DENSITY VECTOR IN Y                                  IDENY=1                          -
CS          FINE MESH DENSITY VECTOR IN Z                                  IDENZ=1                          -
CS          ADAPTIVE WEIGHTED DIAMOND PARAMS                               ALWAYS                          -
CS          RADIUS MODIFIERS IN X                                           IEVT=4.AND.IXM=1                -
CS          RADIUS MODIFIERS IN Y                                           IEVT=4.AND.IYM=1                -
CS          RADIUS MODIFIERS IN Z                                           IEVT=4.AND.IZM=1                -
CS          SN ORDER BY GROUP                                               IGRPSN=1                          -
CS          TIME-DEPENDENT DUMP TIMES                                       IFLXDP.GT.0                      -
CS          TIME-DEPENDENT CROSS SECTION                                    NXTM.GT.0                          -
CS          MODIFERS                                                       -
CS          TIME-DEPENDENT SOURCE MODIFIERS                                NSTM.GT.0                          -
CS          MESA COLLAPSE ENERGY WEIGHTS                                   IMSHCEW.NE.O                      -
CS          MESH COLLAPSE CELL REGION BOUNDARIES                          IMSHCXR.NE.O                      -
CS          DELAYED FISSION SPECTRA                                         ICHID.NE.O                          -
CS          SINGLE CHI ARRAY (FISSION SPECTRA)                             INCHI=1                          -
CS          ***** (REPEAT FOR ALL ZONES)                                 -
CS          * CHI ARRAY (FISSION SPECTRA)                                   INCHI=2                          -
CS          *****                                                       -
CS          QUADRATURE WEIGHTS                                               IQUAD=3                          -
CS          QUADRATURE COSINES                                               IQUAD=3                          -
CS          ***** (REPEAT FOR ALL MOMENTS)                               -
CS          * SOURCE SPECTRUM                                               IQOPT=1                          -
CS          * SOURCE SPATIAL VECTOR(S)                                       IQOPT=2                          -
CS          * ***** (REPEAT FOR ALL GROUPS)                             -
CS          * * SOURCE POINTWISE FULL DISTRIBUTION                         IQOPT=3                          -
CS          * *****                                                       -
CS          * SOURCE SPECTRUM                                               IQOPT=4                          -
CS          * SOURCE SPATIAL VECTOR(S)                                       IQOPT=4                          -

```

```

CS      *          SOURCE SPECTRUM                      IQOPT=6      -
CS      *          SOURCE POINTWISE FULL DISTRIBUTION IQOPT=6      -
CS      *****
CS      ***** (REPEAT FOR ALL GROUPS)                  -
CS      * ***** (REPEAT FOR ALL FACES IN THE ORDER LEFT, RIGHT, -
CS      * *          BOTTOM, TOP, FRONT, BACK, X=L,R,B,T,F,K BELOW) -
CS      * *          BOUNDARY ISOTROPIC SOURCE           IQX=-1      -
CS      * *          BOUNDARY ANISOTROPIC SOURCE         IQX=+1      -
CS      * *          BOUNDARY SOURCE SPECTRUM            IQX=+2      -
CS      * *          BOUNDARY SOURCE SPACE VECTOR 1     IQX=+2      -
CS      * *          BOUNDARY SOURCE SPACE VECTOR 2     IQX=+2      -
CS      * *          BOUNDARY SOURCE ANGLE VECTOR        IQX=+2      -
CS      * *          BOUNDARY SOURCE SPECTRUM            IQX=+3      -
CS      * *          BOUNDARY SOURCE SPACE ARRAY         IQX=+3      -
CS      * *          BOUNDARY SOURCE ANGLE VECTOR        IQX=+3      -
CS      * *          BOUNDARY SOURCE SPECTRUM            IQX=+4      -
CS      * *          BOUNDARY SOURCE SPACE-ANGLE ARRAY  IQX=+4      -
CS      * *          BOUNDARY SOURCE SPECTRUM            IQX=+5      -
CS      * *          BOUNDARY SOURCE ANGLE-SPACE ARRAY  IQX=+5      -
CS      * *****
CS      *****
C
C-----
C-----
CR      FILE IDENTIFICATION                               -
C
CL      HNAME, (HUSE(I), I=1,2), IVERS                   -
C
CW      1+3*MULT=NUMBER OF WORDS                          -
C
CD      HNAME          HOLLERITH FILE NAME - SOLINP - (A6) -
CD      HUSE(I)        HOLLERITH USER IDENTIFICATION (A6) -
CD      IVERS          FILE VERSION NUMBER                 -
CD      MULT           DOUBLE PRECISION PARAMETER         -
CD                   1- A6 WORD IS SINGLE WORD           -
CD                   2- A6 WORD IS DOUBLE PRECISION WORD -
C
C-----
C-----
CR      TITLE CARD COUNT                                  -
C
CL      NHEAD                                                  -
C
CW      1=NUMBER OF WORDS                                     -
C
CD      NHEAD          NUMBER OF TITLE CARDS TO FOLLOW      -
C-----
C-----
CR      TITLE CARD                                         -
C
CC      PRESENT IF NHEAD.GT.0                               -
C
CL      (TITLE(I), I=1,12)                                  -
C
CW      12=NUMBER OF A6 WORDS                               -
C
C-----
C-----
CR      PRINT PARAMETERS                                    -
C
CL      IDPRNT                                              -
C

```

```

CW      30      -
C      -
CL      HCPRNT  -
C      -
CW      30*MULT -
C      -
C-----
C      -
C-----
CR      SPATIAL DIMENSION -
C      -
CL      IDIMEN  -
C      -
CW      1=NUMBER OF WORDS -
C      -
CD      IDIMEN=1 FOR ONEDANT -
C      -
C-----
C-----
CR      RAW CONTROLS AND DIMENSIONS -
C      -
CW      200=NUMBER OF WORDS -
C      -
CD      1 IEVT   TYPE OF CALCULATION -
CD      2 ITH    0/1 - DIRECT/ADJOINT CALCULATION -
CD      3 ISCT   LEGENDRE ORDER OF SCATTERING -
CD      4 ISN    ANGULAR QUADRATURE ORDER -
CD      5 IQUAD  SOURCE OF QUADRATURE SET -
CD      6 ISTART FLUX GUESS FLAG (ZERO FOR ONEDANT) -
CD      7 ICSM   0/1 - NO/YES IN-SOLVER MIXING(FROM ASSIGN= ) -
CD      8 INCHI  0/1/2 - NONE/ONE CHI/ZONWISE CHI -
CD      9 IBL    0/1/2/3 - LEFT BDRY CONDITION -
CD      10 IBR   0/1/2/3 - RIGHT BDRY CONDITION -
CD      -
CD      11 IDENX 0/1/2 - NONE/FINE MESH DENSITY FACTORS BY XMesh/ -
CD      FINE MESH DENSITY FACTORS FOR EVERY MESH -
CD      12 IPVT  0/1/2 - NONE/K-EFF/ALPHA PARAMETRIC EIGENVALUE TYPE -
CD      13 I2ANG 0/1 - NO/YES DO 2 ANGLE SLAB CALCULATION -
CD      14 IQOPT 0/1/2/3/4/5/6 - INHOMOGENEOUS SOURCE OPTION -
CD      15 IQAN  INHOMOGENEOUS SOURCE LEGENDRE ORDER -
CD      16 IQL   -1/0/1/2/3/4/5 LEFT BOUNDARY SOURCE OPTION -
CD      17 IQR   -1/0/1/2/3/4/5 RIGHT BOUNDARY SOURCE OPTION -
CD      18 OITM  OUTER ITERATION LIMIT -
CD      19 IITL  EARLY INNER ITERATION LIMIT -
CD      20 IITM  NEAR CONVERGENCE INNER ITERATION LIMIT -
CD      -
CD      23 FLUXP 0/1/2 - NONE/ISOTROPIC/ALL MOMENTS FLUX PRINT -
CD      24 XSECTP 0/1/2 - NONE/PRINCIPAL/ALL CROSS SECTION PRINT -
CD      25 FISSRP 0/1 - NO/YES FISSION RATE PRINT -
CD      26 SOURCP 0/1/2/3 - NO/AS READ/NORMALIZED/BOTH SOURCE PRINT -
CD      27 GEOMP 0/1 - NO/YES FINE MESH GEOMETRY PRINT -
CD      28 ANGP  0/1 - NO/YES ANGULAR FLUX PRINT -
CD      30 IRMFLX WRITE CODE-DEPENDENT ZONE FLUXES -
CD      -
CD      31 IGRPSN GROUP DEP SN ORDERS (GRPSN) READ IN -
CD      32 RAFLUX 0/1 - NO/YES WRITE ANGULAR FLUX FILE RAFLUX -
CD      33 ISBEDO ALBEDO OPTION -
CD      34 IBALP  0/1 - NO/YES PRINT BALANCES BY ZONE -
CD      35 IANGFLX ANGULAR FLUXES TO DISC -
CD      36 IBB    0/1/2/3/4 - BOTTOM BDRY CONDITION -
CD      37 IBT    0/1/2/3/4 - TOP BDRY CONDITION -
CD      39 IQT    -1/0/1/2/3/4/5 TOP BOUNDARY SOURCE OPTION -
CD      40 IQB    -1/0/1/2/3/4/5 BOTTOM BOUNDARY SOURCE OPTION -
CD      -
CD      41 IXM    0/1 - NO/YES RADIAL MODIFIERS FOR X -

```

CD 42	IYM	0/1 - NO/YES RADIAL MODIFIERS FOR Y	-
CD 43	IZM	0/1 - NO/YES RADIAL MODIFIERS FOR Z	-
CD 44	IDENY	0/1 - NO/FINE MESH DENSITY FACTORS BY YMESH	-
CD 45	IDENZ	0/1 - NO/FINE MESH DENSITY FACTORS BY YMESH	-
CD 50	IBF	0/1/2/3/4 - FRONT BDRY CONDITION	-
CD			-
CD 51	IBK	0/1/2/3/4 - BACK BDRY CONDITION	-
CD 52	NN		-
CD 53	IFLXMOM		-
CD 54	JSFRNT		-
CD 55	JSBACK		-
CD			-
CD 65	IQF	-1/0/1/2/3/4/5 FRONT BOUNDARY SOURCE OPTION	-
CD 66	IQK	-1/0/1/2/3/4/5 BACK BOUNDARY SOURCE OPTION	-
CD			-
CD 73	NOSIGF	0/1 - NONE/SET NUSGF ZERO FOR SOURCE PROBLEMS	-
CD 74	IPLANT	0/1 - NONE/PLANET VARIABLE PRESENT	-
CD 75	JSRITE	0/i = NO/YES WRITE RIGHT BOUNDARY FLUX AT i	-
CD 76	JSBOTT	0/i = NO/YES WRITE BOTTOM BOUNDARY FLUX AT i	-
CD 77	JSTOP	0/i = NO/YES WRITE TOP BOUNDARY FLUX AT i	-
CD 78	JSLEFT	0/i = NO/YES WRITE LEFT BOUNDARY FLUX AT i	-
CD			-
CD 79	EIGONLY	0/1 NO/YES ONLY CONVERGE EIGENVALUE AND FISSIONS	-
CD 86	FCNRAY	NO. OF RAY TRACINGS/BATCH (RAY TRACE OPTION)	-
CD 87	FCNTR	NUMBER OF BATCHES IN THE RAY TRACE OPTION	-
CD 88	NODAL	0/1/2 DD OR AWDD/CL/LL NODAL SPATIAL DIFFERENCING	-
CD 89	NPEY		-
CD 90	IFLXDP		-
CD 91	NXTM		-
CD 92	NSTM		-
CD 93	IAVTAR	0/1 NO/YES WRITE THE AVATAR FILE	-
CD 94	IGREYA	0/1 NO/YES	-
CD 95	LNKMMIX	0/1/2 NO/LNK3DNT/ALNK3DNT	-
CD 96	LNKNMXSP	NUMBER OF MIXING INSTRUCTIONS ON LINK FILE	-
CD 97	NPEZ		-
CD 98	ICHID		-
CD 99	KPRINT		-
CD 100	INCOL		-
CD 101	IMSHCEW		-
CD 102	IMSHCXR		-
CD 103	NO COLAPS-VOID		-
CD 104	MSHCPR		-
CD 105	LSSN		-
CD 106	LSXS		-
CD 107	NCHUNK		-
CD 108	TSA_ITS		-
CD 109	TSA_SN		-
CD 110	WGTDIA		-
CD 111	ISRCH		-
CD 112	FCRSTRT		-
CD 113	FCSN		-
CD 114	TIMACC		-
CD 115	IBFSIZE		-
CD 116	FCSEED		-
CD 117	TIMIITL		-
CD 118	FCWCPO		-
CD 119	MSHFACT		-
CD 120	FCTERM		-
C			-
C			-
C			-
CR		RAW FLOATING DATA	-
C			-
CW		200*MULT=NUMBER OF WORDS	-
C			-

CD	1	EV	EIGENVALUE GUESS	-
CD	2	NORM	NORMALIZATION CONSTANT	-
CD	3	EPSO	OUTER ITERATION CONVERGENCE CRITERION	-
CD	4	EPSI	INNER ITERATION CONVERGENCE CRITERION	-
CD	5	BHGT	BUCKLING HEIGHT	-
CD	6	BWTH	BUCKLING WIDTH	-
CD	7	EVM	EIGENVALUE MODIFIER	-
CD	8	PV	PARAMETRIC VALUE	-
CD	9	XLAL	LAMBDA LOWER LIMIT FOR SEARCHES	-
CD	10	XLAH	LAMBDA UPPER LIMIT FOR SEARCHES	-
CD				-
CD	11	XLAX	SEARCH CONVERGENCE CRITERION	-
CD	12	POD	PARAMETER OSCILLATION DAMPER	-
CD				-
CD	26	EFACT		-
CD	27	TO		-
CD	28	TS		-
CD	29	EOM		-
CD	34	DELTMIN		-
CD				-
CD	36	FCWCO	WEIGHT CUTOFF FOR FIRST COLLISION RAYS	-
CD	40	XPOS		-
CD	41	YPOS		-
CD	42	ZPOS		-
CD	43	BSLA1		-
CD	44	BSLA2		-
CD	45	DXPOS		-
CD	46	DYPOS		-
CD	47	DZPOS		-
CD	48	DPOANG		-
CD	49	DAZANG		-
CD	51	BEFF		-
CD	52	DRADIUS		-
CD	53	DDEPTH		-
CD	55	ASIGT		-
CD	56	BNUSF		-
CD	57	RLWBND		-
CD	58	MAX_SRC_SLOPE		-
CD	60	DELTI		-
CD	61	RDMPNME		-
CD	62	TSA_EPSI		-
CD	63	TSA_BETA		-
CD				-
CD	91	EXTRAS	VECTOR USED BY INDIVIDUAL SOLVERS	-
C				-
C				-
C				-
CR			DEFAULTED CONTROLS AND DIMENSIONS	-
C				-
CN			THIS RECORD IS THE SAME FORMAT AS THE RAW CONTROLS AND DIMENSION	-
CN			RECORD ABOVE, BUT IT CONTAINS THE DEFAULTED VALUES FOR EACH	-
CN			VARIABLE	-
C				-
C				-
C				-
C				-
CR			DEFAULTED FLOATING DATA	-
C				-
CN			THIS RECORD IS THE SAME FORMAT AS THE RAW FLOATING DATA	-
CN			RECORD ABOVE, BUT IT CONTAINS THE DEFAULTED VALUES FOR EACH	-
CN			VARIABLE	-
C				-
C				-
C				-
C				-

```

C
C-----
CR          FINE MESH DENSITY FACTORS
C
CC          PRESENT IF IDENX.EQ.2
C
CL          (DEN(I), I=1, IT) ----NOTE STRUCTURE BELOW----
C
CW          IT*MULT=NUMBER OF WORDS
C
CS          DO 1 K=1, KT
CS          DO 1 J=1, JT
CS          1 READ (N) *LIST AS ABOVE*
C
CD          IT          NUMBER OF FINE MESH INTERVALS
C
C-----
C
C-----
CR          FINE MESH DENSITY VECTOR IN X
C
CC          PRESENT IF IDENX.EQ.1
C
CL          (DEN(I), I=1, IT)
C
CW          IT*MULT=NUMBER OF WORDS
C
CD          IT          NUMBER OF FINE MESH INTERVALS IN X DIRECTION
C
C-----
C
C-----
CR          FINE MESH DENSITY VECTOR IN Y
C
CC          PRESENT IF IDENY.EQ.1
C
CL          (DEN(J), J=1, JT)
C
CW          JT*MULT=NUMBER OF WORDS
C
CD          JT          NUMBER OF FINE MESH INTERVALS IN Y DIRECTION
C
C-----
C
C-----
CR          FINE MESH DENSITY VECTOR IN Z
C
CC          PRESENT IF IDENZ.EQ.1
C
CL          (DEN(K), K=1, KT)
C
CW          KT*MULT=NUMBER OF WORDS
C
CD          KT          NUMBER OF FINE MESH INTERVALS IN Z DIRECTION
C
C-----
C
C-----
CR          AWDD PARAMETERS
C
CC          PRESENT ALWAYS
C
CL          (WDAMP(G), G=1, IGM)

```

```

C -
CW IGM*MULT=NUMBER OF WORDS -
C -
CD IGM NUMBER OF ENERGY GROUPS -
C -
CL (THRSHD(G),G=1,IGM) -
C -
CW IGM*MULT=NUMBER OF WORDS -
C -
CD IGM NUMBER OF ENERGY GROUPS -
C -
C-----
C -
C-----
CR FINE MESH DENSITY FACTORS -
C -
CC PRESENT IF IDENX.EQ.2 -
C -
CL (DEN(I),I=1,IT)----NOTE STRUCTURE BELOW---- -
C -
CW IT*MULT=NUMBER OF WORDS -
C -
CS DO 1 K=1,KT -
CS DO 1 J=1,JT -
CS 1 READ (N) *LIST AS ABOVE* -
C -
CD IT NUMBER OF FINE MESH INTERVALS -
C -
C-----
C -
C-----
CR FINE MESH DENSITY VECTOR IN X -
C -
CC PRESENT IF IDENX.EQ.1 -
C -
CL (DEN(I),I=1,IT) -
C -
CW IT*MULT=NUMBER OF WORDS -
C -
CD IT NUMBER OF FINE MESH INTERVALS IN X DIRECTION -
C -
C-----
C -
C-----
CR FINE MESH DENSITY VECTOR IN Y -
C -
CC PRESENT IF IDENY.EQ.1 -
C -
CL (DEN(J),J=1,JT) -
C -
CW JT*MULT=NUMBER OF WORDS -
C -
CD JT NUMBER OF FINE MESH INTERVALS IN Y DIRECTION -
C -
C-----
C -
C-----
CR FINE MESH DENSITY VECTOR IN Z -
C -
CC PRESENT IF IDENZ.EQ.1 -
C -
CL (DEN(K),K=1,KT) -
C -
CW KT*MULT=NUMBER OF WORDS -
C -

```

```

CD      KT      NUMBER OF FINE MESH INTERVALS IN Z DIRECTION      -
C                                             -
C-----
CR      RADIAL MODIFIER IN X                                       -
C                                             -
CC      PRESENT IF IEVT.EQ.4 .AND. IXM.EQ.1                       -
C                                             -
CL      (RM(I) , I=1, IM)                                          -
C                                             -
CW      IM*MULT=NUMBER OF WORDS                                    -
C                                             -
CD      IM      NUMBER OF COARSE MESH INTERVALS                   -
C                                             -
C-----
C-----
CR      RADIAL MODIFIER IN Y                                       -
C                                             -
CC      PRESENT IF IEVT.EQ.4 .AND. IYM.EQ.1                       -
C                                             -
CL      (RM(I) , J=1, JM)                                          -
C                                             -
CW      JM*MULT=NUMBER OF WORDS                                    -
C                                             -
CD      JM      NUMBER OF COARSE MESH INTERVALS                   -
C                                             -
C-----
C-----
CR      RADIAL MODIFIER IN Z                                       -
C                                             -
CC      PRESENT IF IEVT.EQ.4 .AND. IZM.EQ.1                       -
C                                             -
CL      (RM(K) , K=1, KM)                                          -
C                                             -
CW      KM*MULT=NUMBER OF WORDS                                    -
C                                             -
CD      KM      NUMBER OF COARSE MESH INTERVALS                   -
C                                             -
C-----
C-----
CR      SN ORDER BY GROUP                                          -
C                                             -
CC      PRESENT IF IGRPSN=1                                        -
C                                             -
CL      (GRPSN(IG) , IG=1, IGM)                                    -
C                                             -
CW      IGM*MULT=NUMBER OF WORDS                                    -
C                                             -
CD      IGM      NUMBER OF ENERGY GROUPS                         -
C                                             -
C-----
C-----
CR      TIME-DEPENDENT DUMP TIMES                                  -
C                                             -
CC      PRESENT IF IFLXDP.GT.O                                     -
C                                             -
CL      (TFLXDP(I) , I=1, IFLXDP)                                  -
C                                             -
C-----

```

```

CW   IFLXDP*MULT = NUMBER OF WORDS           -
C                                         -
CL   (NFLXDP(I), I = 1, IFLXDP)             -
C                                         -
CW   IFLXDP=NUMBER OF WORDS                 -
C                                         -
C                                         -
C-----
C-----
CR           TIME-DEPENDENT CROSS SECTION MODIFIERS -
C                                         -
CC           PRESENT IF NXTM.GT.O           -
C                                         -
CL   (XTIMES(I), I=1, NXTM)                -
C                                         -
CW   NXTM*MULT=NUMBER OF WORDS             -
C                                         -
CL   (XAMP(I), I=1, NXTM)                  -
C                                         -
CW   NXTM*MULT = NUMBER OF WORDS           -
C                                         -
C-----
C-----
CR           TIME-DEPENDENT SOURCE MODIFIERS -
C                                         -
CC           PRESENT IF NSTM.GT.O           -
C                                         -
CL   (STIMES(I), I=1, NSTM)                -
C                                         -
CW   NSTM*MULT=NUMBER OF WORDS             -
C                                         -
CL   (SAMP(I), I=1, NSTM)                  -
C                                         -
CW   NSTM*MULT = NUMBER OF WORDS           -
C                                         -
C-----
C-----
CR           MESH COLLAPSE ENERGY WEIGHTS -
C                                         -
CC           PRESENT IF IMSHCEW.NE.O       -
C                                         -
CL   (MSHCEWT(I), I=1, NGROUP)            -
C                                         -
CW   NGROUP*MULT=NUMBER OF WORDS          -
C                                         -
C-----
C-----
CR           MESH COLLAPSE CELL REGION BOUNDARIES -
C                                         -
CC           PRESENT IF IMSHCXR.NE.O       -
C                                         -
CL   (MSHCXRG(I), I=1, 6)                 -
C                                         -
CW   6=NUMBER OF WORDS                     -
C                                         -
C-----
C-----
CR           DELAYED FISSION SPECTRA        -
C                                         -

```

```

CC      PRESENT IF ICHID.NE.0      -
C      -
CL      (CHID_GRP(S) , I=1,NGROUP) -
C      -
CW      NGROUP*MULT=NUMBER OF WORDS -
C      -
C-----
C      -
C-----
CR      CHI - FISSION SPECTRA      -
C      -
CC      PRESENT IF INCHI.NE.0      -
C      -
CL      (CHI (N) , N=1,NGROUP)     -
C      -
CW      NGROUP*MULT=NUMBER OF WORDS -
C      -
C-----
C      -
C-----
CR      QUADRATURE WEIGHTS          -
C      -
CC      PRESENT IF IQUAD.EQ.3      -
C      -
CL      (WGT (M) , M=1,MM)         -
CW      MM*MULT=NUMBER OF WORDS    -
C      -
CD      MM      NUMBER OF ANGLES IN QUADRATURE SET -
C      -
C-----
C      -
C-----
CR      QUADRATURE COSINES          -
C      -
CC      PRESENT IF IQUAD.EQ.3      -
CL      (NLL (N) , N=1, ISN/2)     -
CL      (WGT (M) , M=1, MM)        -
C      -
CL      (MU (M) , M=1,MM)          -
C      -
CL      (ETA (M) , M=1,MM)         -
C      -
CW      MM*MULT=NUMBER OF WORDS    -
C      -
C-----
C      -
C-----
CR      SOURCE SPECTRUM             -
C      -
CC      PRESENT IF IQOPT.EQ.1 .OR. IQOPT.EQ.4 .OR. IQOPT.EQ.6 -
C      -
CL      (SOURCE (N) , N=1,NGROUP)   -
C      -
CW      NGROUP*MULT=NUMBER OF WORDS -
C      -
C-----
C      -
C-----
CR      SOURCE SPATIAL VECTOR IN FIRST DIMENSION -
C      -
CC      PRESENT IF IQOPT.EQ.2 .OR. IQOPT.EQ.4 -
C      -
CL      (SOURCX (I) , I=1, IT)      -

```

```

C
CW   IT*MULT=NUMBER OF WORDS
C
C-----
C
C-----
CR   SOURCE SPATIAL VECTOR IN SECOND DIMENSION
C
CC   PRESENT IF IQOPT.EQ.2 .OR. IQOPT.EQ.4 .AND. IDIMEN.GE.2
C
CL   (SOURCY(J),J=1,JT)
C
CW   JT*MULT=NUMBER OF WORDS
C
C-----
C
C-----
CR   SOURCE SPATIAL VECTOR IN THIRD DIMENSION
C
CC   PRESENT IF IQOPT.EQ.2 .OR. IQOPT.EQ.4 .AND. IDIMEN.EQ.3
C
CL   (SOURCZ(K),K=1,KT)
C
CW   KT*MULT=NUMBER OF WORDS
C
C-----
C
C-----
CR   SOURCE POINTWISE FULL DISTRIBUTION
C
CC   PRESENT IF IQOPT.EQ.3 .OR. IQOPT.EQ.6
C
CL   (SOURCX(I),I=1,IT)----NOTE STRUCTURE BELOW----
C
CW   IT*MULT=NUMBER OF WORDS
C
CS   DO 1 K=1,KT
CS   DO 1 J=1,JT
CS   1 READ(N) *LIST AS ABOVE*
C
C-----
C
C-----
CR   BOUNDARY ISOTROPIC SOURCE (LEFT SHOWN)
C
CC   PRESENT IF IBL.EQ.-1 .AND. IDIMEN.EQ.1 .AND. GROUP.EQ.1
C
CL   (SILEFT(N),N=1,NGROUP)
C
CW   NGROUP*MULT=NUMBER OF WORDS
C
CD   SILEFT(N)          LEFT BOUNDARY SOURCE FOR GROUP N
C
C   ---   ---   ---   ---   ---   ---   ---   ---   ---
C
CC   PRESENT IF IBL.EQ.-1 .AND. IDIMEN.GT.1
C
CL   (SILEFT(J),J=1,JT)----NOTE STRUCTURE BELOW----
C
CW   JT*MULT=NUMBER OF WORDS
C
CS   DO 1 K=1,KT
CS   1 READ(N) *LIST AS ABOVE*
C

```

```

CD      SILEFT(J)          LEFT BOUNDARY SOURCE FOR THE GROUP AT MESH -
CD                                INTERVAL (J,K) -
CD      JT                NUMBER OF FINE MESH INTERVALS IN THE SECOND -
CD                                DIMENSION. EQUALS 1 FOR ONEDANT. -
CD      KT                NUMBER OF FINE MESH INTERVALS IN THE THIRD -
CD                                DIMENSION. EQUALS 1 FOR ONEDANT AND TWODANT. -
C -
C-----
C-----
CR      BOUNDARY ANISOTROPIC SOURCE (LEFT SHOWN) -
C -
CC      PRESENT IF IBL.EQ.+1 -
C -
CL      (SALEFT(M) ,M=1,MMHALF) ----NOTE STRUCTURE BELOW---- -
C -
CW      MMHALF*MULT=NUMBER OF WORDS -
C -
CS      DO 1 K=1,KT -
CS      DO 1 J=1,JT -
CS      1 READ(N) *LIST AS ABOVE* -
C -
CD      SALEFT(M)          LEFT BOUNDARY ANGULAR SOURCE DISTRIBUTION FOR -
CD                                THE GROUP AT INTERVAL (J,K) -
CD      MMHALF            NUMBER OF INCOMING ANGLES IN THE SOURCE -
C -
C-----
C-----
CR      BOUNDARY SOURCE SPECTRUM (LEFT SHOWN) -
C -
CC      PRESENT IF IQL.GE.2 .AND. IQL.LE.4 .AND. GROUP.EQ.1 -
C -
CL      (SOURCE(N) ,N=1,NGROUP) -
C -
CW      NGROUP*MULT=NUMBER OF WORDS -
C -
C-----
C-----
CR      BOUNDARY SOURCE SPATIAL DISTRIBUTION VECTOR 1 (LEFT SHOWN) -
C -
CC      PRESENT IF IQL.EQ.2 .AND. GROUP.EQ.1 -
C -
CL      (SOURCY(J) ,J=1,JT) -
C -
CW      JT*MULT=NUMBER OF WORDS -
C -
C-----
C-----
CR      BOUNDARY SOURCE SPATIAL DISTRIBUTION VECTOR 2 (LEFT SHOWN) -
C -
CC      PRESENT IF IQL.EQ.2 .AND. GROUP.EQ.1 -
C -
CL      (SOURCZ(K) ,K=1,KT) -
C -
CW      KT*MULT=NUMBER OF WORDS -
C -
C-----
C-----
CR      BOUNDARY SOURCE SPACE ARRAY (LEFT SHOWN) -
C -
CC      PRESENT IF IQL.EQ.3 .AND. GROUP.EQ.1 -

```

```

C -
CL (SOURCY(J),J=1,JT) ----NOTE STRUCTURE BELOW---- -
C -
CW JT*MULT=NUMBER OF WORDS -
C -
CS DO 1 K=1,KT -
CS 1 READ(N) *LIST AS ABOVE* -
C -
C----- -
C -
C----- -
CR BOUNDARY SOURCE SPACE-ANGLE ARRAY (LEFT SHOWN) -
C -
CC PRESENT IF IQL.EQ.4 .AND. GROUP.EQ.1 -
C -
CL (SOURCY(J),J=1,JT) ----NOTE STRUCTURE BELOW---- -
C -
CW JT*MULT=NUMBER OF WORDS -
C -
CS DO 1 M=1,MMHALF -
CS DO 1 K=1,KT -
CS 1 READ(N) *LIST AS ABOVE* -
C -
C----- -
C -
C----- -
CR BOUNDARY SOURCE ANGLE-SPACE ARRAY (LEFT SHOWN) -
C -
CC PRESENT IF IQL.EQ.5 .AND. GROUP.EQ.1 -
C -
CL (SALEFT(M),M=1,MMHALF) ----NOTE STRUCTURE BELOW---- -
C -
CW MMHALF*MULT=NUMBER OF WORDS -
C -
CS DO 1 K=1,KT -
CS DO 1 J=1,JT -
CS 1 READ(N) *LIST AS ABOVE* -
C -
C----- -
C -
CEOF -

```

UCFLUX

The UCFLUX file is a binary, code-dependent interface file containing the necessary information for a restart calculation when using the ray tracing first collision option in PARTISN.

```
C*****-
C                                     DATE 02/09/95 -
C                                     -
CF                                UCFLUX -
CE                                CODE DEPENDENT UNCOLLIDED FLUX RESTART FILE -
C                                     -
C*****-
```

```
C-----
C                                     -
C                                     -
CN      THIS FILE ALLOWS A RESTART WHEN USING THE FCSRC OPTION -
C                                     -
C-----
```

```
C-----
CS      FILE STRUCTURE -
CS      RECORD TYPE          PRESENT IF -
CS      ===== -
CS      INT SPECIFICATIONS    ALWAYS -
CS      FP SPECIFICATIONS     ALWAYS -
CS      NUMBER OF TRACKS      I2.GT.0 (TWODANT ONLY) -
CS      INTEGRAL FLUXES       I2.GT.0 (TWODANT ONLY) -
CS      -
CS      ***** (REPEAT FOR NSGRP GROUPS) -
CS      *          UNCOLLIDED FLUX MOMENTS    ALWAYS -
CS      ***** -
C      -
CC      NSGRP IS THE NUMBER OF GROUPS WITH AN INHOMOGENEOUS -
CC      SOURCE -
C      -
C-----
```

```
C-----
CR      INT SPECIFICATIONS          (1D RECORD) -
C      -
CL      NHSTOT, NXYSORE, IPREVTR, NWDS -
C      -
CW      4 =NUMBER OF WORDS -
C      -
CD      NHSTOT          NUMBER OF RAYS TRACED -
CD      NXYSORE        NUMBER OF RAYS TERMINATED -
CD      IPREVTR        NUMBER OF PREVIOUS TRIALS -
CD      NWDS           TOTAL NUMBER OF WORDS IN FILE -
C      -
C-----
```

```

C-----
CR          FP SPECIFICATIONS          (1D RECORD)          -
C
CL      WTMIN,WTMAX,WTTOT,PTIME          -
C
CW      4 =NUMBER OF WORDS          -
C
CD      WTMIN          MINIMUM RAY WEIGHT          -
CD      WTMAX          MAXIMUM RAY WEIGHT          -
CD      WTTOT          TOTAL RAY WEIGHT          -
CD      PTIME          CPU TIME          -
C-----

```

```

C-----
CR          NUMBER OF TRACKS          (2D RECORD)          -
C
CC          PRESENT IF I2.GT.0 (TWO DANT ONLY)          -
C
CL      ((NTENTR(I,J),I=1,IT),J=1,JT)          -
C
CW      IT*JT =NUMBER OF WORDS          -
C
CD      NTENTR(I,J)          NUMBER OF RAYS ENTERING EACH FINE MESH -
C-----

```

```

C-----
CR          INTEGRAL FLUXES          (3D RECORD)          -
C
CC          PRESENT IF I2.GT.0 (TWO DANT ONLY)          -
CC          ALWAYS PRESENT FOR PARTISN          -
C
CL      ((FLRT(I,J,K),I=1,IT),J=1,JT),K=1,KT),          -
CL      ((FLRT2(I,J,K),I=1,IT),J=1,JT),K=1,KT)          -
C
CW      IT*JT*2 =NUMBER OF WORDS          -
C
CD      FLRT(I,J,K)          INTEGRAL FLUXES FOR VARIANCE CALC          -
CD      FLR2T(I,J,k)          INTEGRAL FLUXES**2 FOR VARIANCE CALC          -
C
C      KT=1 FOR TWO DANT          -
C-----

```

```

C-----
CR          LEAKAGES          (1D RECORD)          -
C
CL      (FCRL(I),I=1,IGM),(FCLL(I),I=1,IGM),(FCTL(I),I=1,IGM),          -
CL      (FCBL(I),I=1,IGM),(FCFL(I),I=1,IGM)*,(FCKL(I),IGM)*          -
C
CW      IGM*NF =NUMBER OF WORDS          -
C
CD      FCRL(I)          FCSRC RIGHT LEAKAGE (UNNORMALIZED)          -
CD      FCLL(I)          FCSRC LEFT LEAKAGE (UNNORMALIZED)          -
CD      FCTL(I)          FCSRC TOP LEAKAGE (UNNORMALIZED)          -
CD      FCBL(I)          FCSRC BOTTOM LEAKAGE (UNNORMALIZED)          -
CD      FCFL(I)          FCSRC FRONT LEAKAGE (UNNORMALIZED)*          -
CD      FCKL(I)          FCSRC BACK LEAKAGE (UNNORMALIZED)*          -
C-----

```

```

C          NF=4 FOR TWODANT, 6 FOR PARTISN          -
C          * - PRESENT FOR PARTISN ONLY           -
C-----
C-----
CR          UNCOLLIDED FLUX MOMENTS (4D RECORD)    -
C          -                                     -
CL          (( (FLUC(I,J,K,N), I=1, IT), J=1, JT), K=1, KT), N=1, NM) -
C          -                                     -
CW          IT*JT*KT*NM =NUMBER OF WORDS          -
C          -                                     -
CD          FLUC(I,J,K,N)          UNCOLLIDED FLUX MOMENTS MULTIPLIED BY -
CD          VOLUME, UNNORMALIZED                  -
C          -                                     -
C          KT=1 FOR TWODANT                        -
C          -                                     -
C-----

```




OVERWRITTEN INPUT FILES

On occasion, an input file has the same name as a normal output file. For instance, the RTFLUX file may be input, but a file named RTFLUX containing the fluxes is always written by the SOLVER module at the completion of the run.

Such a file will be overwritten without comment if it is in the local filespace. PARTISN only writes to the local filespace. For a list of the files that PARTISN might write, the user is referred to Table 9.1, “Files Read and Written,” on page 9-10 in chapter “ONEDANT, TWODANT, TWOHEX, TWODANT/GQ, and PARTISN — Code Structure”.

The user can protect input cross-section files through use of a search path name. See the User’s Guide for a discussion of how to do this.

REFERENCES

1. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
2. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physics Codes, Version III," Los Alamos Scientific Laboratory report LA-5486-MS (February 1974).

FILES DESCRIPTIONS CHAPTER INDEX

F

File description

AAFLXM for PARTISN	15-25
AAFLXM for TWODANT	15-22
ADJMAC	15-29
AMFLUX	15-32
ARBFLUX	15-17
ASGMAT	15-34
AZMFLX	15-37
BXSLIB	15-39
EDITIT	15-41
EDTOGX	15-13
EDTOUT	15-7
FISSRC	15-48
GEODST, extended	15-50
GEOSING	15-62
LNK3DNT	15-63
MACRXS	15-66
RAFLXM for PARTISN	15-72
RAFLXM for TWODANT	15-69
RMFLUX	15-76
RZMFLX	15-78
SNXEDT	15-80
SOLINP	15-83
UCFLUX	15-98

Files

Detailed descriptions of	15-1
Input that will be overwritten	15-101

O

Overwriting input files	15-101
-------------------------------	--------

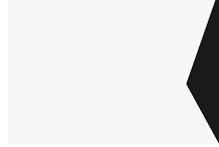
CODE ABSTRACT

Transport Methods Group, CCS-4
Los Alamos National Laboratory

12

CCS-4 — Transport Methods Group





INTRODUCTION

This chapter contains the PARTISN code abstract.

CODE ABSTRACT FOR PARTISN: A CODE PACKAGE FOR MULTI- DIMENSIONAL, TIME-DEPENDENT, DIFFUSION-ACCELERATED, NEUTRAL-PARTICLE TRANSPORT

by

Raymond E. Alcouffe, Randal S. Baker, Jon A. Dahl, and Scott A. Turner
MS D409, Los Alamos National Laboratory
Los Alamos, NM 87545
rsb@lanl.gov

ABSTRACT

1. Program Name and Title: PARTISN (PARallel, Time-Dependent SN)
2. Computer for Which Program is Designed and Other Machine Versions Available: The current release is designed for UNIX-like systems. The specific computers supported fall into two categories - long word and short word. The program has been implemented on the long word Cray J90 and T90 computers. It has also been implemented on Linux, SGI, IBM RS6000, HP9000, and Compaq Alpha short word workstations. The workstation versions use double precision arithmetic. The program has been run in parallel on clusters of SGI

workstations (ASCI Blue Mountain), the IBM SP2 (ASCI Blue Pacific), and Compaq Alphas (ASCI Q). The ASCI Red machine at Sandia is also supported.

3. Problem Solved: PARTISN solves the linear Boltzmann transport equation for neutral particles using the deterministic (SN) method. Both the static (fixed source or eigenvalue) and time-dependent forms of the transport equation are solved in forward or adjoint mode. Vacuum, reflective, periodic, white, or inhomogeneous boundary conditions are solved. General anisotropic scattering and inhomogeneous sources are permitted. PARTISN solves the transport equation on orthogonal (single level or block-structured AMR) grids in 1-D (slab, two-angle slab, cylindrical, or spherical), 2-D (X-Y, R-Z, or R-T) and 3-D (X-Y-Z or R-Z-T) geometries.

4. Method of Solution: PARTISN numerically solves the multigroup form of the neutral-particle Boltzmann transport equation. The discrete-ordinates form of approximation is used for treating the angular variation of the particle distribution. For curvilinear geometries, diamond differencing is used for angular discretization. The spatial discretizations may be either low-order (diamond difference or Adaptive Weighted Diamond Difference (AWDD)) or higher-order (linear discontinuous or exponential discontinuous). Negative fluxes are eliminated by a local set-to-zero-and-correct algorithm for the diamond case (DD/STZ). Time differencing is Crank-Nicholson (diamond), also with a set-to-zero fixup scheme. Both inner and outer iterations can be accelerated using the diffusion synthetic acceleration method, or transport synthetic acceleration can be used to accelerate the inner iterations. The diffusion solver uses either the conjugate gradient or multigrid method. Chebyshev acceleration of the fission source is used. First-collision source treatment options are provided for the elimination of primary ray effects in fixed-source calculations. The angular source terms may be treated either via standard PN expansions or Galerkin scattering. An option is provided for strictly positive scattering sources. Parallelization is performed via a 2-D spatial decomposition, which retains the ability to invert the source iteration equation in a single sweep.

5. Restrictions on the Complexity of the Problem: The code is thoroughly variably dimensioned, with memory requirements determined from the input parameters. Out-of-core (i.e., disk) storage capability options are provided for the flux moments and time-dependent angular fluxes.

6. Typical Running Time: Running time on a single processor is directly related to problem size and to central processor and data transfer speeds. On a SGI R10000, a four-group eigenvalue calculation of an X-Y-Z model of the Fast Test Reactor (FTR) took 9 seconds. The calculation used transport corrected P0 cross sections, an S8 angular quadrature, DD/STZ spatial differencing, and a 14x14x30 spatial mesh. Running time on parallel platforms is sensitive to the

latency and topology of the interconnect, as well as the single processor performance.

7. Unusual Features of the Program: PARTISN is modularly structured in a form that separates the input and output (edit) functions from the main calculational (solver) section of the code. The code makes use of binary, sequential data files, called interface files, to transfer data between modules. Standard interface files whose specifications have been defined by the Reactor Physics Committee on Computer Code Coordination are accepted, used, and created by the code. A free-field card-image input capability is provided for the user. The code provides the user with considerable flexibility in using both card-image or sequential file input and in controlling the execution of modules.

8. Related and Auxiliary Programs: PARTISN is the evolutionary successor to DANTSYS. User input is very similar to that of the DANTSYS code. An auxiliary code, Fracnbox, is available to generate geometry link files for PARTISN from combinatorial geometry input.

9. Status: The current PARTISN release is 3.45. Updates are made approximately three to four times a month.

10. Hardware Requirements: The virtual machine memory must be large enough for the problem being executed. On many architectures, stack size limits must be large enough to allow the placement of temporary arrays on the stack.

11. Programming Languages: The program is written in ANSI standard F90 with a few C language routines used to interface to the Unix operating system. Parallelization is performed using MPI 1.1.

12. Operating System: The program is designed to port to any UNIX-like operating system with minimal effort. Where available, POSIX routines are used to obtain the machine name, cross section path, and access rights. Otherwise, system-specific routines must be used.

13. Other Programming or Operating Information or Restrictions: In addition to compilers, program building requires GNUmake (Version 3.74 or later), GNU-

awk (Version 3.0 or later), and cpp. A Readme file in the top program directory contains complete build instructions.

14. Name and Affiliation of Author or Contributor: Raymond E. Alcouffe, Randal S. Baker, Jon A. Dahl, and Scott A. Turner, Transport Methods Group, Los Alamos National Laboratory.

15. Material Available: The PARTISN source code (~90,000 lines), although export-controlled, is available through RSICC.

BIBLIOGRAPHY

Transport Methods Group, CCS-4
Los Alamos National Laboratory

13

CCS-4 — Transport Methods Group



BIBLIOGRAPHY

1. T. E. Albert and P. Nelson, "Computation of Azimuthally Dependent Albedo Data by Invariant Embedding," in Proc. of Sixth Intl. Conf. on Radiation Shielding, May 16-20, 1983, Tokyo, Vol. I, pp. 283-293.
2. R. E. Alcouffe, "A Diffusion Accelerated S_N Transport Method for Radiation Transport on a General Quadrilateral Mesh", *Nucl. Sci Eng.*, **105**, 191-197, 1990.
3. R. E. Alcouffe, R. D. O'Dell, and F. W. Brinkley, Jr., "A First-Collision Source Method That Satisfies Discrete S_n Transport Balance," *Nucl. Sci. Eng.* **105**, 198 (1990).
4. R. E. Alcouffe, "An Adaptive Weighted Diamond Differencing Method for Three-Dimensional XYZ Geometry," *Trans. Am. Nuc. Soc.* **68**, Part A, 206 (1993).
5. R. E. Alcouffe, "Diffusion Synthetic Acceleration Methods for the Diamond-Difference Discrete-Ordinates Equations," *Nucl. Sci. Eng.* **64**, 344 (1977).
6. R. E. Alcouffe, "The Multigrid Method for Solving the Two-Dimensional Multigroup Diffusion Equation," Proc. Am. Nucl. Soc. Top. Meeting on Advances in Reactor Computations, Salt Lake City, Utah, March 28-31, 1983, Vol. 1, pp 340-351.
7. American National Standard Programming Language Fortran, ANSI X3.198-1992, American National Standards Institute, Inc., New York, NY 10018.
8. G. I. Bell and S. Glasstone, "Discrete Ordinates and Discrete S_N Methods," in Nuclear Reactor Theory, (Van Nostrand Reinhold, New York, 1970), Chap. 5, pp. 232-235.
9. B. G. Carlson and K. D. Lathrop, "Transport Theory-Method of Discrete Ordinates," in Computing Methods in Reactor Physics, H. Greenspan, C. N. Kelber and D. Okrent, Eds. (Gordon and Breach, New York, 1968), Chap. III, p. 185.
10. B. M. Carmichael, "Standard Interface Files and Procedures for Reactor Physics Codes, Version III," Los Alamos Scientific Laboratory report LA-5486-MS (February 1974).
11. K. L. Derstine, "DIF3D: A Code to Solve One-, Two-, and Three-Dimensional Finite-Difference Diffusion Theory Problems," Argonne National Laboratory report ANL-82-64 (April 1984).
12. W. W. Engle, Jr., "A USER'S MANUAL FOR ANISN, A One Dimensional Discrete Ordinates Transport Code With Anisotropic Scattering," Union Carbide report K-1693, (March 1967).
13. D. R. Ferguson and K. L. Derstine, "Optimized Iteration Strategies and Data Management Considerations for Fast Reactor Finite Difference Diffusion Theory Codes," *Nucl. Sci. Eng.* **64**, 593 (1977).
14. R. D. O'Dell, "Standard Interface Files and Procedures for Reactor Physics Codes, Version IV," Los Alamos Scientific Laboratory report LA-6941-MS (September 1977).
15. R. D. O'Dell and R. E. Alcouffe, "Transport Calculations for Nuclear Analysis: The-

- ory and Guidelines for Effective Use of Transport Codes,” Los Alamos National Laboratory report LA-10983-MS (September 1987).
16. R. D. O’Dell, F. W. Brinkley Jr., D. R. Marr, R. E. Alcouffe, “Revised User’s Manual for ONEDANT: A Code Package for One-Dimensional, Diffusion-Accelerated, Neutral-Particle Transport,” Los Alamos National Laboratory manual LA-9184-M, Rev. (December 1989).
 17. W. A. Rhoades and R. L. Childs, “An Updated Version of the DOT4 One- and Two-Dimensional Neutron/Photon Transport Code,” Oak Ridge National Laboratory report ORNL-5851, (July 1982).
 18. W. A. Rhoades and F. R. Mynatt, “THE DOT III TWO-DIMENSIONAL DISCRETE ORDINATES TRANSPORT CODE,” Oak Ridge National Laboratory report ORNL-TM-4280, (September 1973).
 19. W. F. Walters, “The TLC Scheme for Numerical Solution of the Transport Equation on Equilateral Triangular Meshes,” Proc. Am. Nucl. Soc. Top. Meeting on Advances in Reactor Computations, Salt Lake City, Utah, March 28-31, 1983, Vol. 1, pp 151-165.

PARTISN GLOBAL INDEX

Transport Methods Group, CCS-4
Los Alamos National Laboratory

14

CCS-4 — Transport Methods Group

A

Acceleration	
Diffusion synthetic(DSA)	8-14
Upscatter(Grey)	3-25
ACCELERATION DISABLED message	3-26
Adaptive weighted diamond differencing	2-59
Adjoint	
Calculation of	3-43
Edit of	2-78, 4-15
Group order in	2-34, 3-43
Albedoes	
Use of	3-28
Alpha(time absorption) search	1-12, 3-39
Angular flux	
File output	2-57
Input of boundary	2-66
Print output	2-57
Anisotropy	
Cross section scattering source	8-23
Approximation	
Discrete ordinates	8-12
Multigroup	8-11
Transverse leakage	3-38
Array	
Definition of	2-19, 5-9
Notation for order	2-23
Notation for size	2-23
ASSIGN	
Example of use	7-11
Input template	2-50

B

Block	
Definition of	2-20, 5-10
Order in input	2-17
Block-I input	
THREEDANT	2-37
Block-II input	
THREEDANT	2-40
Block-III input	
THREEDANT	2-41
Block-IV input	

THREEDANT	2-48
Blocks	
Input order	2-17
Block-V input	
THREEDANT	2-55
Block-VI input	
THREEDANT	2-71
Boltzmann transport equation	8-9
Boundary condition	
Discussion of	3-27
Options	2-55
Boundary source	2-66, 3-33
Buckling	
Simulated leakage	3-38

C

Character names	
In mixing arrays	2-51
Chi	
Choice from MENDF	2-42
Isotope independent input	2-42
Zone dependent	2-58, 3-30
Coarse mesh	
Definition of	1-12, 3-15
Comments	
Embedded in input lines	2-20, 5-10
Convergence	
Behavior in search	3-42
Iterating to	1-12, 3-17
Tests	3-23
Convergence controls	
Special for criticality	2-57
Cross section library	
ASCII, writing from code	6-21, 6-23
Balancing of	2-42
Binary, writing from code	6-12
BXSLIB	6-12
Card image libraries	6-9
Converting to ASCII	2-42
Coupled neutron-gamma libraries	6-17
Details for inputting	6-1
Edit names on MENDF	2-80
Edit positions and names on	2-73
GRUPXS	6-9
ISOTXS	6-9

MACBCD	6-14
MACRXS	6-13
MENDF5	6-14
MENDF5G	6-15
SNXEDT	6-13
Specifying what, where	2-41
Text, format of	2-46
Text, order within	2-47
Text, position in input stream	2-17
Transport correcting	2-58, 3-36
Writing ASCII library from code	2-42
XSLIB	6-9
XSLIBB	6-13
XSLIBE, XSLIBF	6-15

D

Data item	
Character, definition of	2-19, 5-10
Numeric, definition of	2-19, 5-9
Data operators	
Purpose of	2-20, 5-10
Summary table of	2-22
Table of	5-15
Usage form	2-20, 5-10
Delimiters	
In free field input	5-13
Density factors	
In edits	2-72
In the flux calculation	2-58
Diffusion synthetic acceleration(DSA)	
General method	8-14
Discrete ordinates approximation	8-12
Discrete ordinates equation	
In one dimension	8-29
Discretization of the spatial variable	
In one dimension	8-35
In triangular geometry	8-55
In two dimensions	8-39
In TWOHEX	8-55
Divergence operator in one dimension	8-21
Documentation available	
For THREEDANT	2-13

E

Edit Module	
Function of	9-19
Edit names on MENDF	2-80
Edit positions and names	
Table of	2-73
Edits	
Energy specifications	2-75, 4-11
Reaction rates	
From cross sections	2-72, 4-11
From user defined response functions	2-74, 4-13
Spatial specifications	2-71, 4-9
Energy groups	
Broad groups in edits	2-75, 4-11
Order in adjoint run	2-34, 3-43
Error	
Implementation	10-13
Messages	10-5
Execution of code	
Multiple runs	9-27
On UNIX systems	2-105
Piecewise	9-21
Expansion of the inhomogeneous source	8-28
Expansion of the scattering source	8-23
Extraneous source, see Source, inhomogeneous	

F

FIDO	5-19
File description	
AAFLXM for THREEDANT	11-25
AAFLXM for TWODANT	11-22
ADJMAC	11-29
AMFLUX	11-32
ARBFLUX	11-17
ASGMAT	11-34
AZMFLX	11-37
BXSLIB	11-39
EDITIT	11-41
EDTOGX	11-13
EDTOUT	11-7
FISSRC	11-48
GEODST, extended	11-50
GEOSING	11-62

LNK3DNT	11-63
MACRXS	11-66
RAFLXM for THREEDANT	11-72
RAFLXM for TWODANT	11-69
RMFLUX	11-76
RZMFLX	11-78
SNXEDT	11-80
SOLINP	11-83
UCFLUX	11-98
Files	
Detailed descriptions of	11-1
Input that will be overwritten	11-101
Output, control of	2-57, 2-78, 4-15
Standard interface	9-7
Suppressing writing	2-39, 9-22
Used in mixing	7-17
Where read, written	9-10
Fine mesh	
Definition of	1-11, 3-15
Fine mesh mixing option	2-38
First collision source options	
Ray tracing	2-69
Fission fraction	
Choice from MENDF	2-42
Isotope independent input	2-42
Zone dependent	2-58, 3-30
Fixed field format	5-19
Flow, control	1-16
Flow, data	1-16
Flux	
Input guess from file	2-61
Moments file output	2-57
Normalization of	2-58, 3-36
Print control	2-57
Free field input	
Delimiters	5-13
Summary	2-19
Syntax details	5-13
Terminators	5-13
User specified format	5-16

G

Geometry	
Coarse mesh	3-15
Fine mesh	3-15

Input arrays	2-40
Symmetries treated in ONEDANT	8-19
Grey acceleration	3-25
Group collapse	
In edits	2-75, 4-11
Group dependent quadrature	2-60

H

Highlights of the run	10-15
-----------------------------	-------

I

Implementation Error	10-13
Inner iteration convergence	3-23
Input instructions	
Block order	2-17
Rules for use of	2-33
Input Module	
Function of	9-17
Isotope	
Concept/Definition of	2-49
Iteration	
Monitor print of	3-25
Monitor, warning message from	3-26
Strategy of	1-12, 3-17
Iteration controls	
Eigenvalue calculations	2-56
Of searches	2-61
Source calculations	2-56
Iteration procedure	
General method	8-14

M

Marking on input arrays	
Optional marking	2-33
Required marking	2-33
Materials	
Concept/Definition of	2-49
MATLS	
Example of use	7-9
Input template	2-49
Memory requirements	3-12
Message	

Error	10-5
Implementation Error	10-13
Warning from iteration monitor	3-26
Warning from Run Highlights	10-15
Methods	
Chapter on solution methods	8-1
Diffusion synthetic	8-14
Monte Carlo/Discrete Ordinates	8-43
MINI-MANUAL	
Definition and purpose	2-23
Graphic of	2-24
Mixing	
ASSIGN input array example	7-11
Concepts in	2-49
Fine mesh input option	2-38
Materials	2-49
MATLS input array example	7-9
Premixes	2-49
Terminology	7-7
Tutorial	7-1
Using atomic fractions	7-13
Using weight fractions	7-13
While assigning materials to zones	7-11
Modules	
Suppressing execution of	2-39
Monitor	
Iteration, print of	3-25
Monte Carlo/Discrete Ordinates	
Methods	8-43
Multigroup approximation	8-11
Multiple runs	9-27

N

NEG. SOURCE - ACCELERATION DISABLED message	3-27
NEG. SOURCE - TRANSPORT FLUXES BAD message	3-27
Normalization	
Of edit reaction rates	2-77
Of the flux	2-58, 3-36
Of the source rate	2-58, 3-36
Notation	
For expected order within array	2-23
For expected size of array	2-23
Numeric names	
In mixing arrays	2-51

O

Operator	
Divergence operator in one dimension	8-21
Transport, in two dimensions	8-38
Operators	
In input, see also "Data operators"	2-20, 5-10
Summary table of	2-22
Table of	5-15
Optional marking on input arrays	2-33
Output	
Angular flux file	2-57
Angular flux print	2-57
Controls	2-57
Printed, control of contents	2-57
Overwriting input files	11-101

P

Piecewise execution of code	9-21
PREMIX	
Input template	2-51
Purpose of	2-50
Printed output	
Control of contents	2-57

Q

Quadrature points	
Number in one dimensional geometries	8-29
Number in two dimensions	8-38
Ordering in one dimension	8-30, 8-31, 8-32
Starting directions	8-34
Quadrature sets	
Group dependent	2-60
ONEDANT, choices of	3-28
THREEDANT, choices of	2-60

R

Required marking on input arrays	2-33
Response function	
Edits of	2-74, 4-13
Run highlights	10-15

S

Sample problem	
THREEDANT	
Description	2-83
Input	2-85
Output	2-83
Searches	
Concentration	
Mixing arrays required	2-52, 2-53
Solver arrays required	2-63
Convergence behavior	3-42
Dimension, arrays required	2-62
General control of	2-61, 3-39
Time absorption(alpha)	1-12, 3-39
Sn order	
Group dependent	2-60
Solution	
Chapter on solution methods	8-1
Solver Module	
Function of	9-18
Source	
Anisotropy in	8-23
Boundary angular fluxes	2-66, 3-33
First collision	2-69
Inhibit fission multiplication	2-56
Inhomogeneous, input of	3-31
Normalization of	2-58, 3-36
Volumetric, input of	2-64
Spatial discretization	
Adaptive weighted diamond method(AWDD)	8-41
Diamond method	8-40
In one dimension	8-35
In two dimensions	8-39
Spherical harmonics	
Expansion of the inhomogeneous source	8-28
Expansion of the scattering source	8-23
In one dimensional geometries	8-26
In two dimensions	8-37
Number in one dimensional geometries	8-25
Stacked runs	9-27
Storage	
Memory requirements	2-38, 3-12
String	
Definition of	2-20, 5-10
Structure	

Of code	1-16, 9-10
Summing	
Of reaction rates	2-76
Over energy in edits	2-75

T

Terminating run	2-56, 3-24
Terminators	
In free field input	5-13
Theory, see Methods	
Time absorption(alpha) search	1-12, 3-39
Time limit	2-56
Transport correction of cross section library	2-58, 3-36
Transport equation, Boltzmann	8-9
TRANSPORT FLUXES BAD message	3-26
Transport operator in two dimensions	8-38

U

Upscatter acceleration	3-25
User's Guide contents	
THREEDANT briefing	2-13

V

Void, specifying a	1-12, 2-40, 3-16
Volumetric source, input of	2-64, 3-31

W

Warning messages	
From iteration monitor	3-26
From the Run Highlights	10-15

X

XSLIB	
Format of	2-46, 6-9

Z

ZAID isotope name	7-14
Zone	
Assigning a material to	2-49
Concept/Definition of	1-12, 2-49, 3-16
Specifying a void in	1-12, 2-40, 3-16

This report has been reproduced directly from the best available copy.

It is available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831. Prices are available from (615) 576-8401.

It is available to the public from the National Technical Information Service, US Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

Los Alamos
NATIONAL LABORATORY

Los Alamos, New Mexico 87545