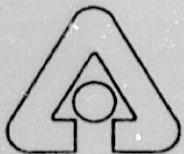


VARIANT: VARlational Anisotropic Nodal Transport for Multidimensional Cartesian and Hexagonal Geometry Calculation

by G. Palmiotti, E. E. Lewis, and C. B. Carrico

Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division



Argonne National Laboratory, Argonne, Illinois 60439
operated by The University of Chicago
for the United States Department of Energy under Contract W-31-109-Eng-38

Reactor Analysis
Division
Reactor Analysis
Division
Reactor Analysis
Division

Argonne National Laboratory, with facilities in the states of Illinois and Idaho, is owned by the United States government, and operated by The University of Chicago under the provisions of a contract with the 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. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Reproduced from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62

Oak Ridge, TN 37831

Prices available from (423) 576-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

ANL-95/40

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

**VARIANT: VARIational Anisotropic Nodal Transport
for Multidimensional Cartesian and Hexagonal Geometry Calculation**

by

G. Palmiotti, E. E. Lewis*, and C. B. Carrico

Reactor Analysis Division
*Northwestern University

BASE TECHNOLOGY

October 1995

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED *ds*

TABLE OF CONTENTS

| | <u>Page</u> |
|---|-------------|
| PROGRAM ABSTRACT | vii |
| ABSTRACT | xi |
| I. INTRODUCTION | 1 |
| II. THE VARIATIONAL NODAL METHOD | 4 |
| II. A The Variational Formulation | 4 |
| II. A.1 The Even-Parity Equations | 4 |
| II. A.2 The Nodal Variational Principle | 6 |
| II. A.3 Nodal Balance | 9 |
| II. B Transport Equation Discretization | 10 |
| II. B.1 The Ritz Procedure | 10 |
| II. B.2 Multigroup Response Matrix Equations | 13 |
| II. C The Spatial and Angular Trial Functions | 14 |
| II. C.1 Spatial Approximations | 14 |
| II. C.2 Angular Approximations | 15 |
| II. C.3 Reflected Boundary Conditions | 17 |
| II. C.4 Vacuum Boundary Conditions | 18 |
| II. D Anisotropic Scattering | 22 |
| II. D.1 Variational Formulation | 22 |
| II. D.2 Within-Group Equations | 24 |
| II. D.3 Multigroup Coupling Equations | 26 |
| II. E Evaluation of Nodal Integrals | 27 |
| III. SOLUTION ALGORITHMS | 30 |
| III. A Red-Black Response Matrix Algorithm | 30 |
| III. B The Partitioned Matrix Algorithm | 32 |
| III. C Inner Iterations | 35 |
| III. D Outer Iterations | 38 |
| IV. NUMERICAL CALCULATIONS | 40 |
| IV. A Two Dimensional Results | 40 |
| IV. B Three Dimensional Results | 43 |
| V. USER INFORMATION | 47 |
| V. A Data Management | 47 |
| V. B Variational Nodal Parameters | 48 |
| V. B.1 Nodal Spatial Approximation | 48 |
| V. B.2 Angular Approximations | 49 |
| V. B.3 Asymptotic Extrapolation Sentinel | 51 |

TABLE OF CONTENTS (Contd.)

| | Page |
|---|-------------|
| V. B.4 Anisotropic Scattering Order (NPNO) and Extended Transport Approximation (NXTR) | 51 |
| V. B.5 Nodal Coupling Coefficient Packing Option | 52 |
| V. B.6 Radial Inner Iteration Algorithm | 52 |
| V. C Limitations | 53 |
| V. D Programming Information | 53 |
| ACKNOWLEDGMENTS | 53 |
| REFERENCES | 59 |
| APPENDIX A Mathematical Scripts Used to Generate the Orthogonal Polynomials and the Submatrices needed to Calculate the Response Matrix Coefficients and the Flux Reconstruction Arrays | 62 |
| APPENDIX B Description of File COMPXS | 115 |
| APPENDIX C Description of Variant Output File NHFLUX | 119 |
| APPENDIX D Description of the BCD Input File A.DIF3D | 123 |

LIST OF FIGURES

| | <u>Page</u> |
|---|-------------|
| 1. Three-Dimensional Cartesian Node | 28 |
| 2. Local Nodal Coordinate System | 29 |
| 3. Orientation of the Positive Directions Along the Sides for Hexagonal Geometry | 29 |
| 4. EBR-II Representation for a Three Dimensional Criticality Problem | 45 |
| 5. M Matrix Ranks for Two-Dimensional Geometries | 50 |
| 6. Call Tree for the Main Branches of the VARIANT Option | 55 |

LIST OF TABLES

| | |
|--|----|
| I. Integral Arrays of Spatial and Angular Basis Functions | 28 |
| II. EBR-II x-y Geometry (enhanced transport effect) | 40 |
| III. EBR-II Hexagonal 2-D Geometry | 42 |
| IV. Takeda Benchmark Model 2 x-y-z Geometry Small FBR | 43 |
| V. EBR-II Hexagonal-z Geometry | 44 |
| VI. EBR-II Ring Axially Integrated Power (MWth) | 46 |
| VII. List of Modified Original DIF3D Subroutines, With Name Changes where a V Replaces an N | 54 |
| VIII. List of the New Subroutines that Were Not Part of the Original Version of DIF3D | 54 |
| IX. List of Modified Original DIF3D Subroutines Without Name Changes | 54 |

PROGRAM ABSTRACT

1. Name of Program: VARIANT- a new nodal module for the DIF3D^{1,2} neutronics code
2. Computer for which Program is Designed and Other Machine Version Packages Available: CRAY X-MP, Sun SPARCstations, IBM RS6000 series.
3. Description of Problem Solved: VARIANT solves the multigroup steady-state neutron diffusion and transport equations in two- and three-dimensional Cartesian and hexagonal geometries using variational nodal methods. The transport approximations involve complete spherical harmonic expansions up to order P₅. Eigenvalue, adjoint, fixed source, gamma heating, and criticality (concentration) search problems are permitted. Anisotropic scattering is treated, and although primarily designed for fast reactor problems, upscattering options are also included.
4. Method of Solution: The neutron and transport equations are solved using a variational nodal method³⁻⁷ with one mesh cell (node) per hexagonal assembly (Cartesian geometry node sizes are specified by the user). The nodal equations are derived from a functional incorporating nodal balance, and reflective and vacuum boundary conditions through Lagrange multipliers. Expansion of the functional in orthogonal spatial and angular (spherical harmonics) polynomials leads to a set of response matrix equations relating partial current moments to flux and source moments. The equations are solved by fission source iteration in conjunction with a coarse mesh rebalance acceleration scheme. The inner iterations are accelerated by a partitioned matrix scheme equivalent to a synthetic diffusion acceleration method⁶.
5. Restrictions on the Complexity of the Problems: Problem dimensions are all variable. Enough memory must be allocated to contain all the information for at least one energy group. Flux and source expansions of up to sixth order are allowed. Partial current expansions up to second order are allowed. Angular and scattering expansions of up to P₅ are allowed. The typical limiting factor for a

problem lies in the storage of response matrices for problems involving large numbers of unique node types. For highly heterogeneous problems involving thousands of different node types, calculation and storage of response matrices represents the primary computational cost.

6. Typical Running Time: The times provided apply to a three dimensional isotropic problem for a small LMR with 30⁰ planar symmetry, 9 energy groups, 14 axial mesh planes and 16 rings of hexagons. The problem consisted of 1694 nodes with 24 compositions and 216 unique node types. Each outer iteration required 70 inner iterations (5 groups required 10 inner iterations and 4 groups required 4 inner iterations). The diffusion calculation required 18 outer iterations and the transport calculation required 19 outer iterations. The diffusion calculation iterations used 41 CPU seconds on a CRAY X-MP/14, 47 seconds on an IBM RS6000, and 107 seconds on a SPARC 20/50. The transport calculation for this problem (with a P₃ angular expansion) required 231 seconds on the CRAY X-MP/14, 1046 seconds on an IBM RS6000 and 2183 seconds on a SPARC 20/50.
7. Unusual Features: Variational nodal methods incorporate a number of attractive features. These include a standard hierarchy of space-angle approximation, well behaved small mesh limits, and the absence of both ray effects and artificial diagonal streaming depressions. Dimensionless parts of the response matrices involving integrals in space and angle are pre-computed once using MATHEMATICA for each geometry option. The results are stored in FORTRAN data statements and used to generate response matrix sets for unique nodes (defined by cross section and dimension data) prior to fission source iteration. Anisotropic scattering (up to order P₅) is also available. VARIANT achieves near Monte Carlo accuracy at a fraction of the cost.
8. Related and Auxiliary Programs: VARIANT reads and writes the standard interface files specified the Committee on Computer Code Coordination (CCCC).
9. Status: VARIANT is currently in use on the Reactor Analysis Division network which consists of Sun SPARCstations and IBM RS6000 series workstations. Modules for perturbation calculations, and inhomogeneous nodes are under development.

10. References:

1. K. L. Derstine, "DIF3D: A Code to Solve One-, Two-, and Three-Dimensional Finite Difference Diffusion Theory Problems," ANL-82-64, Argonne National Laboratory (1982).
2. R. D. Lawrence, "The DIF3D Nodal Neutronics Option for Two- and Three-Dimensional Diffusion Theory Calculations in Hexagonal Geometry," ANL-83-1, Argonne National Laboratory (1983).
3. G. Palmiotti, C. B. Carrico, and E. E. Lewis, "Variational Nodal Methods with Anisotropic Scattering," *Nuclear Science and Engineering*, 115, 233-243, 11/93
4. J.Y. Doriath, F. Malvagi, G. Palmiotti, J. M. Ruggieri, C. B. Carrico, E. E. Lewis, and G. Gastaldo, "Variational Nodal Method (VNM) to Solve 3D Transport Equation: Applications to EFR Design," *Proceedings of Mathematical Methods and Supercomputing in Nuclear Applications*, Karlsruhe, Germany, 4/93, I-571
5. C. B. Carrico, E. E. Lewis, and G. Palmiotti, "Three-Dimensional Variational Nodal Transport Methods for Cartesian, Triangular, and Hexagonal Criticality Calculations", *Nuclear Science and Engineering*, 111, pp. 168 - 179, 6/92.
6. C.B. Carrico and E.E. Lewis, "Variational Nodal Solution Algorithms for Multigroup Criticality Problems," *Proc. Int. Topl. Mtg. Advances in Mathematics, Computations and Reactor Physics*, April 28- May 1, 1991, Pittsburgh, Penn.
7. E.E. Lewis, C.B. Carrico, and G. Palmiotti, "Variational Nodal Formulation for the Spherical Harmonics Equations," *Nuclear Science and Engineering*, in press.

11. Machine Requirements: At least 8 Mb of memory are recommended for program and file buffer storage on short word machines like SPARCstations and RS6000 workstations. External data storage must be available for approximately 40 scratch and interface files. Fourteen of these files are random access scratch files (grouped into 6 file groups) and the remainder are sequential access files with formatted or unformatted record types.

12. Programming Languages Used: FORTRAN 77 is used. The program can be executed entirely in FORTRAN. Dynamic memory allocation routines on the workstations are written in C or supplied from host machine libraries.

13. Operating System: No special requirements are made on the operating system.

14. Other Programming or Operating Information or Restrictions: The stand-alone source code contains approximately 120,000 lines of FORTRAN statements (including DIF3D driver and edit routines).

15. Name and Establishment of Author or Contributor:

G. Palmiotti
Reactor Analysis Division
Argonne National Laboratory

E. E. Lewis
Dept. of Mechanical Engineering
Northwestern University

16. Material Available:

A. User's Manual

B. Magnetic Tape Cartridge Containing

1. Source Code.
2. Sample Problem Data Card-Images.
3. Sample Problem Output.
4. Code Dependent BCD and Binary Card Image File Descriptions.

Distribution may be restricted.

17. Keywords: Neutron diffusion theory, neutron transport, multigroup theory, two-dimensional, three-dimensional, neutron flux, interface current, hexagonal geometry, coarse mesh method, variational nodal method, criticality searches, hexagonal configuration, CCCC, anisotropic scattering, spherical harmonics.

18. Sponsor: U.S. Department of Energy, Nuclear Energy Programs under Contract W-31-109-ENG-38.

VARIANT: VARIational Anisotropic Nodal Transport for Multidimensional Cartesian and Hexagonal Geometry Calculation

by

G. Palmiotti, E. E. Lewis, and C. B. Carrico

ABSTRACT

The theoretical basis, implementation information and numerical results are presented for VARIANT (VARIational Anisotropic Neutron Transport), a FORTRAN module of the DIF3D code system at Argonne National Laboratory. VARIANT employs the variational nodal method to solve multigroup steady-state neutron diffusion and transport problems. The variational nodal method is a hybrid finite element method that guarantees nodal balance and permits spatial refinement through the use of hierarchical complete polynomial trial functions. Angular variables are expanded with complete or simplified P_1 , P_3 or P_5 spherical harmonics approximations with full anisotropic scattering capability. Nodal response matrices are obtained, and the within-group equations are solved by red-black or four-color iteration, accelerated by a partitioned matrix algorithm. Fission source and upscatter iterations strategies follow those of DIF3D. Two- and three-dimensional Cartesian and hexagonal geometries are implemented. Forward and adjoint eigenvalue, fixed source, gamma heating, and criticality (concentration) search problems may be performed.

I. INTRODUCTION

VARIANT (VARIational Anisotropic Neutron Transport) is a FORTRAN module of the DIF3D code system¹ at Argonne National Laboratory. It performs multigroup neutron transport calculations in both Cartesian and hexagonal geometries in two and three dimensions. Both forward and adjoint calculations may be performed. Spherical harmonics are employed to treat the angular variables; at present P₁, P₃ and P₅ approximations are implemented in all geometries and include both within-group and group-to-group anisotropic scattering. The spatial dependence of the flux variables is represented by complete polynomials within coarse mesh nodes, and along internode interfaces. Polynomials as high as fourth order for Cartesian and sixth order for hexagonal geometries are implemented.

Solutions of the within-group neutron transport equation are obtained using the variational nodal method, which originated at Northwestern University and has been developed in close collaboration with Argonne National Laboratory.²⁻¹⁰ The defining feature of the method is a variational principle for the even-parity form of the transport equation in which odd-parity Lagrange multipliers along the node interfaces guarantee neutron conservation for each node. The well-founded variational formulation allows computational algorithms to be derived using the classical Ritz procedure: known trial functions in angle and in space are used to approximate the flux variables and obtain sets of linear algebraic equations for each node, with inter-node coupling specified by concomitant continuity conditions. For computations effectiveness a transformation of variables is then employed to reduce the nodal equations to response matrix form.

The systematic use of the Ritz procedure allows well-defined hierarchies of approximations in angle and in space to be generated. Diffusion or P₁ theory is the natural lowest-order angular approximation to arise from the formulation, allowing diffusion calculations to be compared easily to higher-order spherical harmonics solutions. The treatment of the spatial variables parallels hybrid finite element methods. The formalism allows polynomials of increasing degrees to be used in examining spatial truncation errors by p convergence as an alternative to the standard h convergence obtained from mesh refinement. In addition to the standard spherical harmonic hierarchy of angular approximations, VARIANT's variational formulation is also adapted easily to reduced angular and simplified spherical harmonics approximations,^{8,11} thus providing additional flexibility in trade-offs between accuracy and computational cost.

The foregoing approach contrasts significantly with those nodal methods which were first formulated and applied with great success for diffusion theory and then extended to transport theory. They begin with a statement of nodal balance and employ transverse integration procedures to obtain approximate quasi-one-dimensional equations whose solutions provide the necessary auxiliary conditions. While highly successful in obtaining fast, coarse mesh diffusion solutions, these approaches have been confounded to some extent by the complexity of space-angle coupling found in the transport equation. Difficulties have been encountered in going beyond spatially flat interface assumptions and in reconciling the angular approximation within the nodes with those along the interfaces. Such methods provide only one space-angle "transport approximation" and allow neither space-angle refinement to examine truncation error nor straight-forward provisions for reconstructing intranodal flux distributions. Moreover, they provide neither the capability to treat anisotropic scattering nor straight-forward provisions for adjoint calculations. Discrete ordinate nodal methods circumvent some of these shortcomings by using a standard S_N hierarchy of angular approximations, but they have not been developed sufficiently for reactor calculations to evaluate their potential. More extensive discussions of competing nodal transport methods may be found elsewhere.^{12,13}

As a module of the DIF3D code system, VARIANT makes extensive use of other system modules to perform those operations which do not pertain either to the generation of the response matrices or to the within-group solution algorithms. These include node generation, outer iteration on the fission source and its acceleration, input of both geometry and cross section files and output editing. Substantial modifications were made to handle the input of anisotropic scattering cross section, which had not been a part of the original DIF3D code.

A unique feature of VARIANT is the central role played by symbolic manipulation in generating the nodal response matrices. For each new geometry or level of space-angle approximation, the Ritz procedure spawns many - in most cases thousands - of multidimensional integrals over known trial functions. Error-free evaluation of these large arrays of integrals is intractable by hand. However they are easily put in dimensionless form. Thus we utilize symbolic manipulation in the form of the Mathematica software package¹⁴ to automate the analytical evaluation of the integrals. The resulting arrays of numbers are stored as DATA statements in the FORTRAN subroutines which generate the

response matrices. Thus the symbolic manipulation is performed only once for each new geometry or for each new approximation in space or angle which is added to VARIANT.

Since the variational nodal equations are cast in response matrix form, VARIANT is also able to make extensive use of existing coding in the nodal option¹⁵ of the DIF3D. The node numbering and other data handling capability for performing red-black or four color response matrix iterations in Cartesian and hexagonal geometries, respectively is retained in VARIANT. Nodal coding previously developed by R. Lawrence¹⁵⁻¹⁷ also serves as an excellent point of departure from which to implement the partitioned matrix algorithm developed for VARIANT to accelerate the iterative solution of the within-group response matrix equations.

The remainder of this report is organized as follows. In Chapter 2 the variational nodal method is described and the derivation of the response matrix equations presented. Special attention is given to the treatment of boundary conditions and inclusion of anisotropic scattering. In Chapter 3 the response matrix solutions algorithm and the partitioned matrix acceleration techniques are described. In Chapters 4 and 5 respectively numerical examples and user information are presented.

II. THE VARIATIONAL NODAL METHOD

In this Chapter we set forth the theory behind the variational nodal method and derive the linear algebraic equations used in the resulting multigroup response matrix algorithm. For simplicity, in Sections II. A and B we first formulate the problem and discretize the equations assuming isotropic scattering, and make use only of some of the more general properties of the space-angle approximations. In Section II. C we then examine the spherical harmonics approximation, the associated boundary conditions and spatial approximations in more detail. In Section II. D we generalize the variational nodal method to include both within-group and group-to-group anisotropic scattering. Finally, in Section II. E. we present the symbolic manipulation evaluation of the integrals involved in the coupling coefficient calculations.

II. A. The Variational Formulation

In this section we present the variational basis for the computational algorithms which constitute the variational nodal method. We begin with the within-group transport equation with isotropic scattering and sources:

$$[\boldsymbol{\Omega} \cdot \nabla + \sigma(\mathbf{r})]\Psi(\mathbf{r}, \boldsymbol{\Omega}) = \int d\boldsymbol{\Omega}' \sigma_s(\mathbf{r})\Psi(\mathbf{r}, \boldsymbol{\Omega}') + S(\mathbf{r}) \quad , \quad (2.1)$$

where σ is the total cross section, and σ_s is the within-group scattering cross section; Ψ represents the angular flux and S the group source; \mathbf{r} and $\boldsymbol{\Omega}$ are the neutron position and direction of travel. In the following subsections we first rewrite this equation in even parity form, and then set forth the variational principle and its properties. The section concludes with a demonstration of the nodal balance property of the variational principle.

II. A.1 The Even-Parity Equations

The definitions of the even- and odd-parity flux components are

$$\psi(\mathbf{r}, \boldsymbol{\Omega}) = \frac{1}{2} [\Psi(\mathbf{r}, \boldsymbol{\Omega}) + \Psi(\mathbf{r}, -\boldsymbol{\Omega})] \quad (2.2)$$

and

$$\chi(\mathbf{r}, \boldsymbol{\Omega}) = \frac{1}{2} [\Psi(\mathbf{r}, \boldsymbol{\Omega}) - \Psi(\mathbf{r}, -\boldsymbol{\Omega})] \quad (2.3)$$

respectively. To formulate the problem variationally, we first obtain the even-parity equation with isotropic scattering and sources. This is accomplished by first evaluating Eq. 2.1 at $\mathbf{\Omega}$ and at $-\mathbf{\Omega}$ and then adding one half of the results to obtain

$$\mathbf{\Omega} \cdot \nabla \chi(\mathbf{r}, \mathbf{\Omega}) + \sigma(\mathbf{r})\psi(\mathbf{r}, \mathbf{\Omega}) = \sigma_s(\mathbf{r}) \int d\Omega' \psi(\mathbf{r}, \mathbf{\Omega}') + S(\mathbf{r}) . \quad (2.4)$$

Likewise, subtracting the results yields

$$\mathbf{\Omega} \cdot \nabla \psi(\mathbf{r}, \mathbf{\Omega}) + \sigma(\mathbf{r})\chi(\mathbf{r}, \mathbf{\Omega}) = 0 \quad (2.5)$$

The even parity equation,

$$-\mathbf{\Omega} \cdot \nabla \sigma^{-1} \mathbf{\Omega} \cdot \nabla \psi + \sigma \psi = \sigma_s \phi + S , \quad (2.6)$$

is then obtained by using Eq. 2.5 to express the odd-parity flux in terms of ψ as

$$\chi = - \sigma^{-1} \mathbf{\Omega} \cdot \nabla \psi \quad (2.7)$$

and then eliminating it from Eq. 2.4.

The scalar flux is written in terms of the even-parity flux as

$$\phi = \int d\Omega \psi , \quad (2.8)$$

and the current vector in terms of the odd-parity flux as

$$\mathbf{J} = \int d\Omega \mathbf{\Omega} \chi . \quad (2.9)$$

Thus combining Eqs. 2.7 and 2.9, we have

$$\mathbf{J} = - \sigma^{-1} \int d\Omega \mathbf{\Omega} \mathbf{\Omega} \cdot \nabla \psi . \quad (2.10)$$

On reflected boundaries, both even- and odd-parity flux components must meet the angular symmetry conditions. Vacuum boundaries may be shown to reduce to the conditions¹⁸

$$\psi(r, \mathbf{\Omega}) = \pm \chi(r, \mathbf{\Omega}) \quad \mathbf{n} \cdot \mathbf{\Omega} \lesseqgtr 0 \quad (2.11)$$

where \mathbf{n} is the outward normal on the vacuum surface.

II. A.2 The Nodal Variational Principle

The even-parity transport equation may be formulated as a variational principle in terms of a global functional, F , which is a superposition of volume and surface contributions from the v spatial nodes and γ nodal interfaces comprising the problem domain:

$$F[\psi, \chi] = \sum_v F_v[\psi, \chi] , \quad (2.12)$$

where the contribution from node v is

$$F_v[\psi, \chi] = \int_v dV \left\{ \int d\Omega [\sigma^{-1} (\Omega \cdot \nabla \psi)^2 + \sigma \psi^2] - \sigma_s \phi^2 - 2\phi S \right\} + 2 \int_\gamma d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_v \psi \chi \quad (2.13)$$

In the absence of the interface term containing χ , Eqs. 2.12 and 2.13 reduce to the functional first formulated by Vladimirov¹⁹, and since used as the basis for many finite-element and related approximations to the transport equation. The use of χ as a Lagrange multiplier at node interfaces is the unique feature which differentiates this functional from previous even-parity variational formulations and gives rise to the variational nodal method. For as we shall see, the continuity requirements of more conventional spatial finite element approximations are relaxed, while neutron conservation is enforced on each node.

Requiring this functional to be stationary with respect to variations in ψ and χ may be shown to lead to the even-parity Euler-Lagrange equation within each node and the continuity of both even and odd-parity fluxes across the interfaces. This is accomplished as follows.¹⁹ Suppose we let ψ_v be the reference even parity flux for $\mathbf{r} \in V_v$, and χ_γ the corresponding odd parity flux for $\mathbf{r} \in \Gamma_\gamma$. Next, we examine the effect of taking arbitrary variations about the reference functions:

$$\psi = \psi_v + \delta\psi , \quad \mathbf{r} \in V_v \quad (2.14)$$

$$\chi = \chi_\gamma + \delta\chi . \quad \mathbf{r} \in \Gamma_\gamma \quad (2.15)$$

Substituting these variations into Eq. 2.13, we may write

$$F_v[\psi_v + \delta\psi, \chi_\gamma + \delta\chi] = F_v[\psi_v, \chi_\gamma] + \delta F_v[\psi, \chi] + \delta^2 F_v[\psi, \chi] , \quad (2.16)$$

where the three terms on the right are referred to respectively as the zero, first and second variations with respect to ψ and χ . The zero variation is just Eq. 2.13 evaluated with the

reference solution, while the second variation contains only products of the variations, $(\delta\psi)^2$ and $\delta\psi\delta\chi$. Here, the first variation is the focus of interest, since for the functional of Eq. 2.12 to be stationary, the sum of these variations must vanish.

The contribution of node V_v to the first variation may be written explicitly as

$$\begin{aligned} \delta F_{V_v}[\psi, \chi] = & 2 \int_{V_v} dV \left\{ \int d\Omega \left[\sigma^{-1} (\Omega \cdot \nabla \delta\psi) \Omega \cdot \nabla \psi_v + \sigma \psi_v \delta\psi_v \right] - \delta\phi (\sigma_s \phi_v + S) \right\} \\ & + 2 \int_{\gamma} d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_\gamma (\chi_\gamma \delta\psi + \psi_v \delta\chi) \end{aligned} \quad (2.17)$$

To put the first variation in more transparent form, we utilize the identity

$$\sigma^{-1} (\Omega \cdot \nabla \delta\psi) \Omega \cdot \nabla \psi_v = -\delta\psi \Omega \cdot \nabla \sigma^{-1} \Omega \cdot \nabla \psi_v + \nabla \cdot (\Omega \delta\psi \sigma^{-1} \Omega \cdot \nabla \psi_v) \quad , \quad (2.18)$$

along with the divergence theorem,

$$\int_{V_v} dV \int d\Omega \Omega \cdot \nabla (\delta\psi \Omega \cdot \nabla \psi_v) = \int_{\gamma} d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_\gamma \delta\psi \Omega \cdot \nabla \psi_v \quad (2.18a)$$

and

$$\int d\Omega \delta\psi = \delta\phi \quad (2.18b)$$

to rearrange terms and obtain

$$\begin{aligned} \delta F_{V_v}[\psi, \chi] = & 2 \int_{V_v} dV \int d\Omega \delta\psi \left[-\Omega \cdot \nabla \sigma^{-1} \Omega \cdot \nabla \psi_v + \sigma \psi_v - \sigma_s \phi_v - S \right] \\ & + 2 \int_{\gamma} d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_\gamma \delta\psi (\sigma^{-1} \Omega \cdot \nabla \psi_v + \chi_\gamma) + 2 \int_{\gamma} d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_\gamma \psi_v \delta\chi \end{aligned} \quad (2.19)$$

Requiring Eq. 2.12, the global functional, to be stationary is equivalent to requiring the first order variation to vanish:

$$\delta F[\psi, \chi] = \sum_v \delta F_{V_v}[\psi, \chi] = 0 \quad (2.20)$$

Thus, the volume term from each δF_{V_v} must vanish if δF is to vanish. But since $\delta\psi$ within each node is arbitrary, this takes place only if the bracketed term in Eq. 2.19 vanishes. But this term and Eq. 2.6 are identical. Thus within each node, the even-parity transport equation is the functional's Euler-Lagrange equation.

The terms over the internal interfaces must be treated somewhat differently. Consider the interface between nodes V_v and $V_{v'}$. Any such interface, designated by Γ_γ and \mathbf{n}_γ , is opposite another node interface, say $\Gamma_{\gamma'}$ and $\mathbf{n}_{\gamma'}$, such that $\Gamma_{\gamma'} = \Gamma_\gamma$ and $\mathbf{n}_{\gamma'} = -\mathbf{n}_\gamma$. From Eq. 2.19 we see that the contribution of this interface to the variation of Eq. 2.20 may be written as the sum of just two integrals,

$$2 \int_{\gamma} d\Gamma \int d\Omega \mathbf{\Omega} \cdot \mathbf{n}_\gamma \delta\psi (\sigma^{-1} \mathbf{\Omega} \cdot \nabla \psi_v - \sigma^{-1} \mathbf{\Omega} \cdot \nabla \psi_{v'}) + 2 \int_{\gamma'} d\Gamma \int d\Omega \mathbf{\Omega} \cdot \mathbf{n}_{\gamma'} \delta\chi (\psi_v - \psi_{v'}) \quad (2.21)$$

since the $\chi_\gamma \delta\psi$ terms in the second integral of Eq. 2.19 cancel. For the second term to vanish with arbitrary variations, the even parity flux must be continuous across the interface. Likewise, for the first term to vanish the flux gradient terms, which are seen from Eq. 2.7 to represent the odd-parity flux, must also be continuous across the interface. Thus the exact interface conditions are met. Finally, note that discontinuities in the cross sections at the node interface have no effect on the foregoing argument (we would need only to place nodal subscripts on the cross sections).

We have yet to consider the boundary conditions on the outer surface of the problem domain. The functional is not varied on reflective boundaries. Rather, the essential symmetry conditions are imposed on the angular distribution of even- and odd-parity fluxes. This causes the $\psi\chi$ term in Eq. 2.13 to be identically equal to zero on reflected boundaries. On vacuum boundaries, the replacement of the $\psi\chi$ term with the integral

$$\int_{\gamma} d\Gamma \int d\Omega |\mathbf{n}_\gamma \cdot \mathbf{\Omega}| \psi^2 \quad (2.22)$$

yields Eq. 2.11. These are referred to as modified natural boundary conditions. We shall return to a more detailed treatment of boundary conditions.

Before proceeding, we observe that the natural lowest-order angular approximation with the foregoing variational formulation is the diffusion or P₁ approximation. If we require the even-parity flux to be independent of angle, $\psi(\mathbf{r}, \mathbf{\Omega}) \rightarrow \phi(\mathbf{r})$, and likewise take $\chi(\mathbf{r}, \mathbf{\Omega}) \rightarrow 3\mathbf{\Omega} \cdot \mathbf{J}(\mathbf{r})$, the diffusion equation becomes the Euler-Lagrange equation, and continuity of the scalar flux and normal current component across interfaces is imposed by the Lagrange-multiplier term.

II. A.3 Nodal Balance

An important property of the nodal formulation is the imposition of neutron balance over each node. Nodal balance may be demonstrated as follows. Suppose we define the volume-averaged scalar flux for a particular node as

$$\bar{\phi} = \frac{1}{V_v} \int_v \phi(\mathbf{r}) dV \quad (2.23)$$

and write the even parity flux as

$$\psi(\mathbf{r}, \Omega) = \bar{\phi} + \psi_o(\mathbf{r}, \Omega) \quad (2.24)$$

where the second term is required only to be orthogonal to $\bar{\phi}$:

$$\int_v dV \int d\Omega \psi_o(\mathbf{r}, \Omega) = 0. \quad (2.25)$$

Likewise, we define the average source as

$$\bar{S} = \frac{1}{V_v} \int_v S(\mathbf{r}) dV \quad (2.26)$$

and write

$$S(\mathbf{r}) = \bar{S} + S_o(\mathbf{r}) \quad (2.27)$$

with the orthogonality condition

$$\int_v S_o(\mathbf{r}) dV = 0 \quad (2.28)$$

If we insert Eqs. 2.24 and 2.27 into the functional given by Eq. 2.13, and utilize the orthogonality conditions, and the definition of \mathbf{J} , we may rearrange terms to obtain

$$\begin{aligned} F_v[\psi, \chi] = & (\sigma - \sigma_s) V_v \bar{\phi}^2 - 2 V_v \bar{\phi} \bar{S} + 2 \bar{\phi} \int_\gamma d\Gamma \mathbf{n}_\gamma \cdot \mathbf{J} \\ & + \int_v dV \left\{ \int d\Omega \left[\sigma^{-1} (\Omega \cdot \nabla \psi_o)^2 + \sigma \psi_o^2 \right] - \sigma_s \left(\int d\Omega \psi_o \right)^2 \right\} + 2 \int_\gamma d\Gamma \int d\Omega \Omega \cdot \mathbf{n}_\gamma \psi_o \chi \end{aligned} \quad (2.29)$$

Note that only the first three terms contain $\bar{\phi}$. Thus if we let $\bar{\phi} \rightarrow \bar{\phi} + \delta\bar{\phi}$ and require F_v to be stationary with respect to arbitrary variations $\delta\bar{\phi}$, we obtain the nodal balance equation

$$(\sigma - \sigma_s) V_v \bar{\phi} + \int_\gamma d\Gamma \mathbf{n}_\gamma \cdot \mathbf{J} = V_v \bar{S} \quad (2.30)$$

which just states that absorption plus leakage must be equal to the number of source neutrons produced in the node. This proof that nodal balance is preserved whether ψ_0 represents the exact solution or only some approximation thereof. The approximate case is very important, for it states that nodal neutron balance is maintained, independent of the even- and odd-parity flux approximations which are used.

II. B. Transport Equation Discretization

In this section we utilize the foregoing variational formulation to discretize the transport equation and obtain a set of linear equations suitable for efficient numerical computations. Accomplishing this entails choosing a suitable set of space-angle trial functions and employing it in a classical Ritz procedure. The equations which result from the Ritz procedure are then cast in a form suitable for within-group response matrix calculations. These, in turn, are embedded in a multigroup formalism. The choice of trial functions is central to the development of accurate and computationally efficient methods. Generally, we utilize orthogonal polynomials in space and spherical harmonics in angle. We defer, however, a detailed discussion of trial functions and their associated boundary conditions to Section II. D. In this section, we need specify only some of the more general properties of the trial functions necessary to carry out the discretization.

II. B.1 The Ritz Procedure

The classical Ritz procedure consists of approximating the dependent variable or variables in a variational principle with a set of known trial or basis functions, and determining the unknown coefficients by requiring the functional to be stationary with respect to variations in the coefficients. We apply the Ritz procedure by approximating the even- and odd-parity fluxes as separable expansions of spatial and angular trial functions with unknown coefficients. With the convention hereafter that repeated *English* (but not Greek) subscripts imply summation, these take the form:

$$\psi(\mathbf{r}, \Omega) \approx f_i(\mathbf{r}) g_m^\pm(\Omega) \zeta_{im} \quad \mathbf{r} \in V_v \quad (2.31)$$

and

$$\chi(\mathbf{r}, \Omega) \approx h_{j\gamma}(\mathbf{r}) k_{n\gamma}(\Omega) \chi_{jn\gamma} . \quad \mathbf{r} \in \Gamma_\gamma \quad (2.32)$$

Since, at present, only isotropic scattering is considered, the even-parity group source is independent of angle and may be approximated as

$$S(\mathbf{r}) \approx f_i(\mathbf{r})s_i \quad \mathbf{r} \in V_v \quad (2.33).$$

and the scalar flux as

$$\phi(\mathbf{r}) \approx f_i(\mathbf{r})\delta_{1m}\zeta_{im}. \quad \mathbf{r} \in V_v \quad (2.34)$$

In the foregoing equations the ζ_{im} and $\chi_{jn\gamma}$ are arrays of unknown coefficients, and the s_i are source coefficients. The $f_i(\mathbf{r})$ and $h_{j\gamma}(\mathbf{r})$ represent spatial basis functions which are complete polynomials. They are orthonormal over the node volume and surfaces, respectively, meeting the conditions

$$\int_V f_i(\mathbf{r})f_{i'}(\mathbf{r})dV = \delta_{ii'} \quad (2.35)$$

and

$$\int_\gamma h_{j\gamma}(\mathbf{r})h_{j'\gamma}(\mathbf{r})d\Gamma = \delta_{jj'}. \quad (2.36)$$

The angular basis functions, $g_m^+(\Omega)$, within the node are even-parity spherical harmonics meeting the orthonormality conditions

$$\int g_m^+(\Omega)g_{m'}^+(\Omega)d\Omega = \delta_{mm'}. \quad (2.37)$$

The odd-parity basis function, $k_{m\gamma}(\Omega)$, along the interfaces consist of odd-order spherical harmonics; their form is discussed further in section II. C.

Inserting the expansions of Ψ , χ , and S into Eq. 2.13 results in the reduced functional

$$F_V[\zeta_{im}, \chi_{jn\gamma}] = \zeta_{im}A_{ii'}^{mm'}\zeta_{i'm'} - 2\zeta_{im}s_{im} + 2\sum_\gamma \zeta_{im}M_{ij\gamma}^{mn}\chi_{jn\gamma} \quad (2.38)$$

where for convenience we have defined $s_{im} = \delta_{0m}s_i$. The matrices in this equation are defined as

$$A_{ii'}^{mm'} = \sigma^{-1} P_{ii'}^{kl} H_{kl}^{mm'} + V_v \delta_{ii'} (\sigma \delta_{mm'} - \sigma_s \delta_{1m} \delta_{1m'}) \quad (2.39)$$

and

$$M_{ij\gamma}^{mn} = D_{ij\gamma} E_{mny}. \quad (2.40)$$

Each of the elements of these matrices is given in terms of integrals over known spatial or angular trial functions as defined in Table I. The isotropic source moments are given by

$$s_i = \int dV f_i(\mathbf{r}) S(\mathbf{r}) \quad (2.41)$$

The reduced functional may be written in a more compact form by defining ζ and χ_γ as partitioned vectors formed from the successive columns of ζ_{im} and $\chi_{jn\gamma}$, which are the arrays of unknown coefficients. The resulting functional appears as

$$F_V[\zeta, \chi] = \zeta^T \mathbf{A} \zeta - 2\zeta^T \mathbf{s} + 2 \sum_{\gamma} \zeta^T \mathbf{M}_{\gamma} \chi_{\gamma} \quad (2.42)$$

where the partitioning of \mathbf{A} , \mathbf{s} and \mathbf{M} is consistent with that of ζ and χ_γ . We may eliminate the sum over the surfaces by defining a single vector over the surfaces

$$\chi^T = [\chi_1^T, \chi_2^T, \dots, \chi_\gamma^T, \dots] \quad (2.42a)$$

and the corresponding coupling matrix

$$\mathbf{M} = [\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_\gamma, \dots] \quad (2.42b)$$

Equation 2.42 may then be written as

$$F_V[\zeta, \chi] = \zeta^T \mathbf{A} \zeta - 2\zeta^T \mathbf{s} + 2\zeta^T \mathbf{M} \chi \quad (2.42c)$$

Requiring the functional to be stationary with respect to variations in ζ^T then yields

$$\zeta = \mathbf{A}^{-1} \mathbf{s} - \mathbf{A}^{-1} \mathbf{M} \chi \quad (2.43)$$

The variation with respect to χ_γ across an interface leads to the requirement that

$$\psi_\gamma = \mathbf{M}_\gamma^T \zeta \quad (2.44)$$

be continuous across each interface. Thus for the surfaces of the node we form an even-parity vector, whose subvectors are the ψ_γ , in terms of the internal trial function ζ :

$$\Psi = \mathbf{M}^T \zeta \quad (2.44a)$$

Combining Eqs.2.43 and 2.44a, we have

$$\Psi = \mathbf{M}^T \mathbf{A}^{-1} \mathbf{s} - \mathbf{M}^T \mathbf{A}^{-1} \mathbf{M} \chi . \quad (2.45)$$

This equation relates the even-parity flux moments on the node interfaces to the source moments within the node and to the odd-parity flux moments on the node interfaces.

II. B.2 Multigroup Response Matrix Equations

Equation 2.45 may be viewed as a generalization of the T^{-1} form of a within-group response matrix equation, which has previously been developed only for diffusion theory.²¹ To obtain a response matrix in conventional form, we introduce the change of variables

$$\mathbf{j}^{\pm} = \frac{1}{4} \Psi \pm \frac{1}{2} \chi \quad , \quad (2.46)$$

where \mathbf{j}^+ and \mathbf{j}^- are, respectively, outgoing and incoming partial current-like moments, each integrated over the corresponding node surface Γ_{γ} . In the diffusion approximation these reduce to the partial currents. Inverting Eq. 2.46 then yields

$$\Psi = 2(\mathbf{j}^+ + \mathbf{j}^-) \quad (2.46a)$$

and

$$\chi = \mathbf{j}^+ - \mathbf{j}^- . \quad (2.46b)$$

Combining Eqs. 2.45 and 2.46, we may then write the nodal response matrix equation in the form

$$\mathbf{j}^+ = \mathbf{B} \mathbf{s} + \mathbf{R} \mathbf{j}^- , \quad (2.47)$$

where

$$\mathbf{R} = [\mathbf{G} + \mathbf{I}]^{-1} [\mathbf{G} - \mathbf{I}] \quad (2.48)$$

$$\mathbf{B} = [\mathbf{G} + \mathbf{I}]^{-1} \mathbf{C} \quad (2.49)$$

The matrices \mathbf{G} and \mathbf{C} are partitioned into submatrices defined for each interface. The submatrices are defined as

$$\mathbf{G}_{\gamma\gamma} = \frac{1}{2} \mathbf{M}_{\gamma}^T \mathbf{A}^{-1} \mathbf{M}_{\gamma} , \quad (2.50)$$

and

$$\mathbf{C}_{\gamma} = \frac{1}{2} \mathbf{M}_{\gamma}^T \mathbf{A}^{-1} . \quad (2.51)$$

Once the partial current moments are determined, the even-parity flux moments for the node interior can be determined from the coefficients given by Eq. 2.43. Using Eq. 2.46b, we have

$$\boldsymbol{\zeta} = \mathbf{A}^{-1} \mathbf{s} - 2\mathbf{C}(\mathbf{j}^+ - \mathbf{j}^-) , \quad (2.52)$$

where the first subvector ϕ of the vector $\boldsymbol{\zeta}$ contains the scalar flux moments.

The multigroup coupling equation for group g is given in the standard form

$$\mathbf{S} = \frac{\chi_g}{k} \nu \sigma_{fg} \phi_{g'} + \sigma_{gg'} \phi_{g'} , \quad (2.53)$$

where the cross section notation is conventional. With the scalar flux expanded in each group as indicated in Eq. 2.53, we obtain

$$s_{gi} = k^{-1} \delta_{m0} \chi_g \nu \sigma_{fg'} \zeta_{g'i0} + \sigma_{gg'} \zeta_{g'i0} \quad (2.54)$$

where the subscript g is added to the source moments to denote the energy group, and we continue to use the shorthand notation $s_{im} = \delta_{0m} s_i$. The group source moments depend only on the corresponding scalar flux moments, $\zeta_{g'i0}$, in the higher energy groups, $g' < g$.

II. C The Spatial and Angular Trial Functions

We next examine the trial functions in space and angle in more detail. In choosing the level of the spatial approximation, attention must be given to the rank of the matrix which couples the nodal to the interface approximations. Likewise, care must be taken in the coupling of the angular approximations if the classical spherical harmonics equations are to be obtained.

II. C.1 Spatial Approximations

The spatial trial functions are taken to be complete polynomials both within the nodes and along the interfaces. The internal polynomial is taken to be of a higher order than that on the interface. In cases where there are relatively few response matrix types, there is little

penalty in making the interior polynomial of high enough order that it differs little for the exact spatial solution. However, in problems with many response matrix types, the formation of the response matrices, particularly the inversion of the A matrix, becomes expensive as the dimension of the A matrix grows with the number of space-angle trial functions. Therefore an important question is that of determining the lowest reasonable order for the complete polynomial for the node interior.

An important criteria is that the D matrix must have full rank.²² Spatial approximations which result in rank deficient matrices prevent convergence of the red-black response matrix solution algorithms from being carried to completion. Round-off errors introduce extraneous solutions which do not grow, but persist in preventing eigenvalue calculations from being converged beyond the sixth or seventh decimal place.

II. C.2 Angular Approximations

The even- and odd-order spherical harmonics angular trial functions appearing in the \mathbf{g} and \mathbf{k} vectors must be specified with care. Variational nodal methods based on spherical harmonics expansions solve the even-angular-parity flux equations within the nodes, while continuity between nodes is provided by even- and odd-parity flux moments. In three-dimensional odd-order P_N approximations, there are $N(N+1)/2$ coupled second-order differential equations within each node. There are, however, $N(N+1)/2$ even- and $(N+1)(N+2)/2$ odd-parity moments across the interfaces. Thus $N+1$ odd-parity continuity conditions must be eliminated, since additional conditions would result in an over-determined set of nodal equations. To derive general P_N approximations, we turn to the use of the Romyantsev interface conditions.²³ As detailed elsewhere,⁹ the Romyantsev conditions are identical to those imposed by the variational nodal functional, provided the choice of odd-parity trial functions is restricted to those that result in a full rank matrix coupling odd- and even-parity moments. This is accomplished most simply by deleting the $Y_{n\pm n}$ terms from the odd-order interface expansions.⁹

To apply the odd-order P_N approximation to the foregoing functional we expand the even- and odd- parity fluxes in terms of the spherical harmonics defined by

$$Y_{pq}(\Omega) = C_{pq} P_p^q(\mu) \begin{cases} \cos(q\omega) \\ \sin(q\omega) \end{cases}, \quad \begin{matrix} p = 0, 1, 2, \dots, N \\ |q| = 0, 1, 2, \dots, p \end{matrix} \quad (2.55)$$

where $P_p^q(\mu)$ are the associated Legendre polynomials. The coefficients C_{pq} are chosen such that

$$\int Y_{pq}(\Omega) Y_{p'q'}(\Omega) d\Omega = \delta_{pp'} \delta_{qq'} , \quad (2.56)$$

and we follow the convention that $q \geq 0$ signifies the cosine series and $q < 0$ the sine series.

Within the nodes, we approximate the even-parity flux by

$$\psi(\mathbf{r}, \Omega) = \sum_{pq} Y_{pq}(\Omega) \zeta_{pq}(\mathbf{r}) \quad \begin{array}{l} p = 0, 2, 4, \dots, N-1 \\ |q| = 0, 1, 2, \dots, p \end{array} \quad (2.57)$$

At the interfaces we employ the odd-parity flux approximation

$$\chi(\mathbf{r}, \Omega) = \sum_{pq} Y_{pq}(\Omega) \chi_{pq}(\mathbf{r}) , \quad \begin{array}{l} p = 1, 3, 5, \dots, N \\ |q| = 0, 1, 2, \dots, p-1 \end{array} \quad (2.58)$$

where the angles in the odd-parity expansion are defined in terms of \mathbf{n} , the outward normal to the interface. A central point is the deletion of the $Y_{p\pm p}$, $p=1, 3, 5, \dots, N$ terms from the odd-parity expansion of Eq. 2.58. These deletions yield the correct number of odd-parity interface conditions. Equally important, it is demonstrated elsewhere that the resulting spherical harmonics formulation satisfies the Romyantsev interface conditions and results in a full-rank coupling matrices between the even-order spherical harmonics expansions within the nodes and the odd-order expansion at the interfaces.⁹

We may write the variational nodal form of the spherical harmonics equations compactly by first expressing the expansions of Eqs. 2.31 and 2.32 as vector relationships. Define the vector of even-parity angular trial functions $\mathbf{g}_m^+(\Omega)$ as

$$\mathbf{g}(\Omega)^T = \left[Y_{00}, Y_{2,2}, Y_{2,-1}, Y_{20}, Y_{21}, Y_{22}, Y_{4,4}, \dots \right] \quad (2.59)$$

and a corresponding vector consisting of the odd-parity trial functions $\mathbf{k}_m(\Omega)$:

$$\mathbf{k}(\Omega)^T = \left[Y_{10}, Y_{3,2}, Y_{3,-1}, Y_{30}, Y_{31}, Y_{32}, Y_{5,4}, \dots \right] . \quad (2.60)$$

The foregoing conditions are general and may be applied to any odd-order spherical harmonics approximation. In earlier implementations of the variational nodal

method a somewhat different form of the interface conditions were used in P_3 approximations.²⁻⁷ For the three-dimensional P_3 approximation the correct number of odd-parity conditions may be obtained by requiring continuity of the $P_1(\mu)$, $P_1(\eta)$, $P_1(\xi)$ and $P_3(\mu)$, $P_3(\eta)$, $P_3(\xi)$ moments, where μ is the direction cosine perpendicular to and η and ξ parallel to the interface. These moments have lead to consistently accurate numerical results in two- and three-dimensional calculations.

The fortuitous correspondence of the number of required conditions with the number of odd-order Legendre polynomials with direction cosines perpendicular and parallel to the interface, however, holds only for the P_3 approximation. They therefore cannot be extended to P_5 or higher approximations. Moreover they do not satisfy the Rumyantsev conditions and result in an angular coupling matrix which is rank deficient. Unlike rank deficiency in the spatial trial functions, there seems to be no effect on convergence if it appears in the angular variables.

A number of other angular approximations are also be employed within the framework of the variational nodal method to reduce the number of interface basis functions without a commensurate loss of accuracy. The reduced^{8,11} and the simplified spherical harmonics⁸ approximations are discussed elsewhere. With any set of angular trial functions used to approximate the transport equation, one must also specify a compatible set of approximate boundary conditions. Both reflected and vacuum boundary conditions are included in VARIANT. These two classes of conditions are treated somewhat differently, since in the variational formulation, reflected conditions are "essential" and must be imposed on the trial functions, while vacuum conditions are "modified natural" and are incorporated into the variational format through the addition of appropriate surface terms to the functional.

II. C.3 Reflected Boundary Conditions

With the foregoing angular trial functions, the Rumaynstev interface conditions are satisfied by requiring ψ_γ and χ_γ to be continuous across nodal interfaces. The components of the vector χ_γ are the odd-parity expansion coefficients $\chi_{j\gamma}$ along the interface. In contrast, the vector ψ_γ is expressed by Eq. 2.44 as a linear combination of the even-parity coefficients ζ_{im} within the nodes. In expanded form these linear combinations may be written as

$$\psi_{j\gamma} = D_{ij\gamma} E_{im\gamma} \zeta_{ir} \quad (2.61)$$

Reflected boundary conditions are essential in the variational formulations and therefore must be imposed directly on the $\psi_{jn\gamma}$ and $\chi_{jn\gamma}$ coefficients. To do this, we first note that the index n in the $\psi_{jn\gamma}$ and $\chi_{jn\gamma}$ coefficients corresponds to the odd-parity spherical harmonic ordering in the vector $\mathbf{k}(\boldsymbol{\Omega})$ as defined in Eq. 2.60.

As shown elsewhere,⁹ the angular symmetry conditions for reflected boundary conditions are satisfied if the $\psi_{jn\gamma}$ coefficients are set equal to zero for values of n corresponding to Y_{pq} with even q , and the $\chi_{jn\gamma}$ coefficient are set to zero for terms corresponding to odd q . Suppose we partition the vectors of interface vectors according to even and odd values of q :

$$\boldsymbol{\Psi}_\gamma = \begin{bmatrix} \boldsymbol{\Psi}_e \\ \boldsymbol{\Psi}_o \end{bmatrix} \quad \boldsymbol{\chi}_\gamma = \begin{bmatrix} \boldsymbol{\chi}_e \\ \boldsymbol{\chi}_o \end{bmatrix} \quad (2.62)$$

Then the reflective boundary conditions are then

$$\boldsymbol{\Psi}_o = \mathbf{0} \quad , \quad \boldsymbol{\chi}_e = \mathbf{0} \quad \mathbf{r} \in \Gamma_r \quad (2.63)$$

and therefore the scalar product vanishes:

$$\boldsymbol{\Psi}_\gamma^T \boldsymbol{\chi}_\gamma = \mathbf{0} \quad \mathbf{r} \in \Gamma_r \quad (2.64)$$

This result, combined with Eq. 2.44, causes the Lagrange multiplier term $\zeta^T \mathbf{M}_\gamma \boldsymbol{\chi}_\gamma$ to vanish for reflected boundaries from the reduced functional, Eq. 2.42 and from subsequent equations. If Eqs. 2.63 are inserted into Eq. 2.46 for the partial current moments, we obtain on reflected boundaries

$$\mathbf{j}_e^- = \mathbf{j}_e^+ \quad , \quad \mathbf{j}_o^- = -\mathbf{j}_o^+ \quad \mathbf{r} \in \Gamma_r \quad (2.65)$$

which are the conditions employed in the response matrix solution algorithm.

II. C.4 Vacuum Boundary Conditions

Vacuum boundary conditions, in contrast to reflected conditions, are modified natural boundary conditions. By modifying the functional with an appropriate surface term along the vacuum, the exact condition is obtained by requiring the functional to be

stationary along the boundary. There are three ways in which this characteristic of vacuum boundary conditions can be incorporated into the computational algorithm.

The first is the classical approach described in Section II.A. We remove the Lagrange multiplier integral from Eq. 2.13 along the vacuum boundary and replace it with the integral, shown as Eq. 2.22. Then when the functional is required to be stationary with respect to variations $\delta\psi(\mathbf{r},\Omega)$, $\mathbf{r} \in \Gamma_v$, the correct vacuum boundary conditions results. However, it is in terms of the gradient of the even parity flux rather than the odd-parity Lagrange multiplier with which we would like to work. For if the Ritz procedure is applied using this formulation, the partial current moments along the vacuum boundary are eliminated, and the condition is incorporated into a response matrix of reduced dimension. This approach is awkward to apply, gives rise to response matrices which are boundary-condition-dependent and is difficult to incorporated into iterative solution algorithms for the response matrix equations.

The foregoing difficulty is circumvented by retaining the odd-parity Lagrange multiplier on the vacuum boundary as follows. Instead of replacing the vacuum surface term in the functional by Eq. 2.13, we add the following term consisting of two integrals,

$$I = \int_v d\Gamma \int d\Omega |\mathbf{n} \cdot \Omega| \psi^2 - 2 \int_v d\Gamma \int d\Omega \mathbf{n} \cdot \Omega \psi \chi \quad (2.66)$$

to yield

$$F_v[\psi, \chi] = \int_v dV \left[\int d\Omega \sigma^{-1} (\Omega \cdot \nabla \psi)^2 + \sigma \phi^2 \right] - 2 \int_v dV \phi S + 2 \int_v d\Gamma \int d\Omega \Omega \cdot \mathbf{n} \psi \chi \\ \int_v d\Gamma \int d\Omega |\Omega \cdot \mathbf{n}| \psi^2 - 2 \int_v d\Gamma \int d\Omega \Omega \cdot \mathbf{n} \psi \chi \quad (2.67)$$

Requiring the functional to be stationary with respect to $\delta\chi(\mathbf{r},\Omega)$ along the vacuum boundary then yields $\psi(\mathbf{r},\Omega) = \psi''(\mathbf{r},\Omega)$. When combined with this condition, the variation $\delta\psi''(\mathbf{r},\Omega)$ then yields the correct vacuum conditions given by Eq. 2.11.

To apply the Ritz procedure, we approximate ψ'' and χ similarly to Eqs. 2.31 and 2.32 on the vacuum boundary,

$$\psi''(\mathbf{r},\Omega) \approx f_i(\mathbf{r}) g_m^+(\Omega) \zeta_{im}'' \quad \mathbf{r} \in \Gamma_v \quad (2.68)$$

and

$$\chi(\mathbf{r}, \Omega) \approx h_{j\gamma}(\mathbf{r}) k_{n\gamma}(\Omega) \chi_{jn\gamma} \quad \mathbf{r} \in \Gamma_v \quad (2.69)$$

Equation 2.66 then takes the form

$$I = \zeta_{im}'' W_{ii'\gamma} L_{mm'\gamma} \zeta_{i'm'}'' - 2 \zeta_{im}'' D_{ij\gamma} E_{mn\gamma} \chi_{jn\gamma} \quad (2.70)$$

where elements of the L and W arrays are given in terms of the angular and spatial trial functions respectively in Table 1. Defining

$$N_{ii'\gamma}^{mm'} = W_{ii'\gamma} L_{mm'} \quad (2.71)$$

and utilizing the definition of $M_{ij\gamma}^{mn}$, we may rewrite Eq. 2.70 as

$$I = \zeta_{im}'' N_{ii'\gamma}^{mm'} \zeta_{i'm'}'' - 2 \zeta_{im}'' M_{ij\gamma}^{mn} \chi_{jn\gamma} \quad (2.72)$$

Writing this expression in vector form and adding it to the reduced functional given by Eq. 2.42 then yields

$$F_v[\zeta, \chi] = \zeta^T A \zeta - 2 \zeta^T s + 2 \sum_{\gamma} \zeta^T M_{\gamma} \chi_{\gamma} + \zeta''^T N_{\gamma} \zeta'' - 2 \zeta''^T M_{\gamma} \chi_{\gamma} \quad (2.73)$$

Requiring the reduced functional to be stationary with respect to variations $\delta \chi$ yields $\zeta'' = \zeta$ on the vacuum boundary and likewise taking $\delta \zeta''$ yields

$$N_{\gamma} \zeta'' = M_{\gamma} \chi_{\gamma} \quad (2.74)$$

Eliminating ζ'' between these results and solving for ζ yields

$$\zeta = N_{\gamma}^{-1} M_{\gamma} \chi_{\gamma} \quad \mathbf{r} \in \Gamma_v \quad (2.75)$$

Then applying Eq.2.44 to obtain the even-parity surface variable, we obtain the vacuum boundary condition:

$$\psi_{\gamma} = M_{\gamma}^T N_{\gamma}^{-1} M_{\gamma} \chi_{\gamma} \quad (2.76)$$

Finally, if we make the transformation of variables to the partial current moments defined by Eq. 2.46 we have

$$\mathbf{j}^+ = \left(\frac{1}{2} \mathbf{M}_\gamma^T \mathbf{N}_\gamma^{-1} \mathbf{M}_\gamma + \mathbf{I} \right)^{-1} \left(\frac{1}{2} \mathbf{M}_\gamma^T \mathbf{N}_\gamma^{-1} \mathbf{M}_\gamma - \mathbf{I} \right) \mathbf{j}^- \quad (2.77)$$

A somewhat simpler form of the vacuum boundary condition can be obtained by applying the requirement that the reduced functional be stationary only with respect to the angular variables. We begin again with the integrals of Eq. 2.66, but this time we expand the even- and odd-parity fluxes only in angle, allowing arbitrary spatial variation,

$$\psi''(\mathbf{r}, \Omega) \approx g_m^+(\Omega) \zeta_m''(\mathbf{r}) \quad \mathbf{r} \in \Gamma_v \quad (2.78)$$

$$\chi(\mathbf{r}, \Omega) \approx k_{m\gamma}(\Omega) \chi_{m\gamma}(\mathbf{r}) \quad \mathbf{r} \in \Gamma_v \quad (2.79)$$

Equation 2.66 then reduce to a difference of spatial integrals

$$I = \int_v d\Gamma \zeta_m'' L_{mm\gamma} \zeta_m'' - 2 \int_v d\Gamma \zeta_m'' E_{mn\gamma} \chi_{n\gamma} \quad (2.80)$$

Taking the appropriate variations now yields $\zeta_m'' = \zeta_m$ and $L_{mm\gamma} \zeta_m'' = E_{mn\gamma} \chi_{n\gamma}$, which in matrix form may be expressed as

$$\mathbf{L} \zeta = \mathbf{E} \chi \quad \mathbf{r} \in \Gamma_v \quad (2.81)$$

respectively. Therefore solving for ζ and combining the result with Eq.2.44 yields the vacuum boundary condition

$$\psi = \mathbf{E}^T \mathbf{L}^{-1} \mathbf{E} \chi. \quad \mathbf{r} \in \Gamma_v \quad (2.82)$$

It may be shown with Eq. 2.46 that on the vacuum boundaries may be expressed in terms of the partial current moments as

$$\mathbf{j}^- = \left[\frac{1}{2} \mathbf{E}^T \mathbf{L}^{-1} \mathbf{E} + \mathbf{I} \right]^{-1} \left[\frac{1}{2} \mathbf{E}^T \mathbf{L}^{-1} \mathbf{E} - \mathbf{I} \right] \mathbf{j}^+ \quad \mathbf{r} \in \Gamma_v \quad (2.83)$$

Here however Ψ , χ and the corresponding partial current vectors are spatially dependent, meaning that unlike Eq. 2.77 which imposes the condition only on the m spatial moments, here it is imposed at each point \mathbf{r} on the vacuum boundary. This is too strong a condition. It may be relaxed, however, by requiring that Eq. 2.83 hold only for the spatial moments $\int_V d\Gamma h_{j\mathbf{n}\gamma}(\mathbf{r})\Psi_{\mathbf{n}\gamma}(\mathbf{r})$ and $\int_V d\Gamma h_{j\mathbf{n}\gamma}(\mathbf{r})\chi_{\mathbf{n}\gamma}(\mathbf{r})$ and therefore for the corresponding vector of the $j_{\mathbf{n}\gamma}^+$ and $j_{\mathbf{n}\gamma}^-$.

II. D Anisotropic Scattering

With the methodology thus far developed, we are now prepared to generalize the variational nodal method to include anisotropic scattering. The arguments contained in the preceding section concerning trial functions and boundary conditions remain valid. Thus we need only to repeat the operations of Sections II. A and B in generalizing the variational formalism and obtaining the multigroup response matrix equations with anisotropic scattering.⁷ Our starting point is the within-group transport equation with anisotropic scattering:

$$[\boldsymbol{\Omega} \cdot \nabla + \sigma(\mathbf{r})]\Psi(\mathbf{r}, \boldsymbol{\Omega}) = \int d\boldsymbol{\Omega}' \sigma_s(\mathbf{r}, \boldsymbol{\Omega}, \boldsymbol{\Omega}')\Psi(\mathbf{r}, \boldsymbol{\Omega}') + S(\mathbf{r}, \boldsymbol{\Omega}) \quad , \quad (2.84)$$

where σ is the total cross section, and σ_s is the within-group anisotropic scattering cross section; Ψ represents the angular flux and S the anisotropic group source.

II. D.1 Variational Formulation

To formulate the problem variationally, we must obtain the even-parity equation with isotropic scattering and sources. This is accomplished by first using the even-parity flux definitions in Eqs. 2.2 and 2.3 to obtain the following pair of second order equations which are generalizations of Eq. 2.4 and 2.5

$$\boldsymbol{\Omega} \cdot \nabla \chi(\mathbf{r}, \boldsymbol{\Omega}) + \sigma(\mathbf{r})\psi(\mathbf{r}, \boldsymbol{\Omega}) = \int d\boldsymbol{\Omega}' \sigma^+(\mathbf{r}, \boldsymbol{\Omega}, \boldsymbol{\Omega}')\psi(\mathbf{r}, \boldsymbol{\Omega}') + S^+(\mathbf{r}, \boldsymbol{\Omega}) \quad (2.85)$$

and

$$\boldsymbol{\Omega} \cdot \nabla \psi(\mathbf{r}, \boldsymbol{\Omega}) + \sigma(\mathbf{r})\chi(\mathbf{r}, \boldsymbol{\Omega}) = \int d\boldsymbol{\Omega}' \sigma^-(\mathbf{r}, \boldsymbol{\Omega}, \boldsymbol{\Omega}')\chi(\mathbf{r}, \boldsymbol{\Omega}') + S^-(\mathbf{r}, \boldsymbol{\Omega}) \quad , \quad (2.86)$$

where we have divided the anisotropic scattering kernel into even- and odd-parity components, σ^+ and σ^- , each of which is expanded in spherical harmonics of corresponding parity,

$$\sigma^\pm(\mathbf{r}, \Omega \cdot \Omega') = \sigma_m^\pm(\mathbf{r}) g_m^\pm(\Omega) g_m^\pm(\Omega') \quad (2.87)$$

Again repeated English indices denote summation, and the anisotropic cross section components, σ_m , do not contain a factor of $2\ell+1$ (see reference 7 for a discussion of the expansion of the scattering kernel). Combining equations yields

$$\chi = -\sigma^{-1} \Omega \cdot \nabla \psi + \sigma^{-1} S^- - \sigma^{-2} \int d\Omega' \tilde{\sigma}^-(\Omega \cdot \Omega') \left[\Omega' \cdot \nabla \psi(\Omega') - S^-(\Omega') \right], \quad (2.88)$$

where for brevity we have defined

$$\tilde{\sigma}^-(\Omega \cdot \Omega') = \left(1 - \sigma_m^-/\sigma\right)^{-1} \sigma_m^- g_m^-(\Omega) g_m^-(\Omega'). \quad (2.89)$$

Since Eq. 2.88 is an explicit relation for χ in terms of the even-parity flux and group source, we may substitute it into Eq. 2.85 to obtain the within-group even-parity equation with anisotropic scattering included:

$$\begin{aligned} & -\Omega \cdot \nabla \sigma^{-1} \Omega \cdot \nabla \psi - \Omega \cdot \nabla \sigma^{-2} \int d\Omega' \tilde{\sigma}^-(\Omega \cdot \Omega') \Omega' \cdot \nabla \psi(\Omega') + \sigma \psi = \\ & \int d\Omega' \sigma^+(\Omega \cdot \Omega') \psi(\Omega') + S^+ - \Omega \cdot \nabla \sigma^{-1} \left[S^- + \sigma^{-1} \int d\Omega' \tilde{\sigma}^-(\Omega \cdot \Omega') S^-(\Omega') \right]. \end{aligned} \quad (2.90)$$

The variational functional for the even-parity transport equation may be generalized to include anisotropic scattering terms in the foregoing equations. The global functional, F , is a superposition of volume and surface contributions as in Eq. 2.12. With anisotropic scattering, the contribution from node v is

$$\begin{aligned} F_v[\psi, \chi] = & \int_v dV \int d\Omega \left[\sigma^{-1} (\Omega \cdot \nabla \psi)^2 + \sigma \psi^2 \right] \\ & + \int_v dV \left\{ \tilde{\sigma}_m^- \sigma^{-2} \left[\int d\Omega g_m^- \Omega \cdot \nabla \psi \right]^2 - \sigma_m^+ \left[\int d\Omega g_m^+ \psi \right]^2 \right\} - 2 \int_v dV \int d\Omega \psi S^+ \\ & + 2 \int_v dV \int d\Omega \Omega \cdot \nabla \psi \sigma^{-1} \left[S^- + \sigma^{-1} \int d\Omega' \tilde{\sigma}^-(\Omega \cdot \Omega') S^-(\Omega') \right] + 2 \int_\Gamma d\Gamma \int d\Omega \Omega \cdot \mathbf{n} \psi \chi. \end{aligned} \quad (2.91)$$

Requiring the functional to be stationary with respect to variations in ψ yields Eq. 2.90 as the Euler-Lagrange equation within V_v and Eq. 2.88 along the interfaces. The continuity of ψ across the interfaces is assured by requiring the functional to be stationary with respect to interface variations in χ , while the continuity of \mathcal{X} is imposed by Lagrange multiplier terms applied on nodal interfaces.

II. D.2 Within-Group Equations

A Ritz procedure employed to obtain the nodal response matrices for each energy group parallels the isotropic case. We first approximate the even- and odd-parity fluxes as separable expansions of spatial and angular trial functions, as in Eqs. 2.12 and 2.13. The even- and odd-parity group sources are expanded as

$$S^\pm(\mathbf{r}, \Omega) \approx f_i(\mathbf{r}) g_m^\pm(\Omega) s_{im}^\pm, \quad \mathbf{r} \in V_v. \quad (2.92)$$

where the source coefficients are given by

$$s_m^\pm = \int d\Omega g_m^\pm S^\pm. \quad (2.93)$$

Inserting the expansions of Ψ , \mathcal{X} , and S into Eq. 2.91 results in the reduced functional

$$F[\zeta_{im}, \chi_{jn\gamma}] = \zeta_{im} \tilde{A}_{ii'}^{mm'} \zeta_{i'm'} - 2\zeta_{im} s_{im}^+ - 2\zeta_{im} T_{ii'}^{mm'} s_{i'm'}^- + 2 \sum_{\gamma} \zeta_{im} M_{ij\gamma}^{mn} \chi_{jn\gamma} \quad (2.94)$$

This differs from the isotropic scattering case in two respects. First, in the A matrix additional terms augment the isotropic case to account for the within-group anisotropic scattering:

$$\tilde{A}_{ii'}^{mm'} = A_{ii'}^{mm'} + \sigma^{-1} P_{ii'}^{kl} \sigma^{-1} \tilde{\sigma}_p^{-1} V_k^{mp} V_\ell^{m'p} + V_v \delta_{ii'} \sigma_q^+ \delta_{mq} \delta_{m'q} \quad (2.95)$$

Second, the even- and odd-parity source moments:

$$s_{im}^\pm = \int dV f_i(\mathbf{r}) \int d\Omega g_m^\pm(\Omega) S^\pm(\mathbf{r}, \Omega) . \quad (2.96)$$

appear in conjunction with the array

$$\mathbf{T}_{ii'}^{mm'} \equiv (\sigma - \sigma_{m'})^{-1} \mathbf{U}_{ii'}^l \mathbf{V}_l^{mm'} \quad (2.97)$$

which operates on the odd-parity source. These are required to treat the anisotropic group-to-group scattering. The \mathbf{U} and \mathbf{V} matrices, which do not appear with isotropic scattering, are given in terms of the known angular and spatial trial functions. They are included in Table I.

Aside from the division of the group source into even- and odd-parity components the reduced functional is quite similar to its isotropic counterpart given in Eq. 2.42.

$$F_v[\zeta, \chi] = \zeta^T \tilde{\mathbf{A}} \zeta - 2\zeta^T \mathbf{s}^+ - 2\zeta^T \mathbf{T} \mathbf{s}^- + 2 \sum_{\gamma} \zeta^T \mathbf{M}_{\gamma} \chi_{\gamma}. \quad (2.98)$$

Requiring the functional to be stationary with respect to variations in ζ^T then yields

$$\zeta = \tilde{\mathbf{A}}^{-1} \mathbf{s}^+ + \tilde{\mathbf{A}}^{-1} \mathbf{T} \mathbf{s}^- - \sum_{\gamma} \tilde{\mathbf{A}}^{-1} \mathbf{M}_{\gamma} \chi_{\gamma}. \quad (2.99)$$

while the variation with respect to χ_{γ} across an interface again yields Eq. 2.44. Taken together, Eqs. 2.44 and 2.99 yield a result analogous to Eq. 2.45:

$$\psi_{\gamma} = \mathbf{M}_{\gamma}^T \tilde{\mathbf{A}}^{-1} \mathbf{s}^+ + \mathbf{M}_{\gamma}^T \tilde{\mathbf{A}}^{-1} \mathbf{T} \mathbf{s}^- - \sum_{\gamma} \mathbf{M}_{\gamma}^T \tilde{\mathbf{A}}^{-1} \mathbf{M}_{\gamma} \chi_{\gamma}. \quad (2.100)$$

To obtain a response matrix equations, the partial current variables defined by Eq. 2.46 are again introduced. The result is

$$\mathbf{j}^+ = \mathbf{B}^+ \mathbf{s}^+ + \mathbf{B}^- \mathbf{s}^- + \mathbf{R} \mathbf{j}^-, \quad (2.101)$$

where \mathbf{R} is defined as before by Eq. 2.48, while in the source moment terms

$$\mathbf{B}^+ = [\mathbf{G} + \mathbf{I}]^{-1} \mathbf{C} \quad (2.102)$$

and

$$\mathbf{B}^- = \mathbf{B}^+ \mathbf{T} \quad (2.103)$$

For anisotropic scattering, however, \mathbf{A} is replaced by $\tilde{\mathbf{A}}$ in the definitions of the \mathbf{G} and \mathbf{C} matrices. Analogous to Eq. 2.43, the even-parity flux moments for the node interior are given by

$$\zeta = \tilde{\mathbf{A}}^{-1} \mathbf{s}^+ + \tilde{\mathbf{A}}^{-1} \mathbf{T} \mathbf{s}^- - 2\mathbf{C}(\mathbf{j}^+ - \mathbf{j}^-) . \quad (2.104)$$

II. D.3 Multigroup Coupling Equations

In the case where only within-group anisotropic scattering is included, the odd-parity source term vanishes and the even-parity source includes only scalar flux moments from other energy groups. The anisotropic scattering terms thus affect only the magnitudes of the \mathbf{R} and \mathbf{B} matrix elements. With group-to-group anisotropic scattering, however, the source for group g contains both even- and odd-parity components:

$$S^+(\mathbf{\Omega}) = \frac{\chi_g}{k} \nu \Sigma_{fg'} \int d\mathbf{\Omega}' \psi_g(\mathbf{\Omega}) + \int d\mathbf{\Omega}' \sigma_{gg'}^+(\mathbf{\Omega} \cdot \mathbf{\Omega}') \psi_g(\mathbf{\Omega}') \quad (2.105)$$

and

$$S^-(\mathbf{\Omega}) = \int d\mathbf{\Omega}' \sigma_{gg'}^-(\widehat{\mathbf{\Omega}} \cdot \widehat{\mathbf{\Omega}}') \chi_g(\mathbf{\Omega}') , \quad (2.106)$$

To eliminate the odd-parity flux from the group source terms we express the even- and odd-parity components of the group-to-group scattering cross section in terms of the like-parity spherical harmonics g^+ and g^- :

$$\sigma_{gg'}^\pm(\mathbf{\Omega} \cdot \mathbf{\Omega}') = \sigma_{gg'm}^\pm \mathcal{G}_m^\pm(\mathbf{\Omega}) \mathcal{G}_m^\pm(\mathbf{\Omega}') \quad g' < g \quad (2.107)$$

After a fair amount of algebraic manipulation the χ flux components within the nodes can be eliminated, and we obtain for s_{gim}^\pm

$$s_{gim}^+ = k^{-1} \delta_m \chi_g \nu \Sigma_{fg'} \zeta_{g'i0} + \sigma_{gg'm}^+ \zeta_{g'im} \quad (2.108)$$

and

$$s_{gim}^- = (\sigma_{g^-} - \sigma_{g'm}^-)^{-1} \sigma_{gg'm}^- s_{g'im}^- - T_{i'i}^{m'm} \zeta_{g'i'm'} , \quad (2.109)$$

The even-parity group source moments depend only on the corresponding flux moments, $\zeta_{g'im}$, in the higher energy groups, $g' < g$, and on $\zeta_{g'i0}$, the scalar flux moments. The odd-parity source moments, however, are a function of the $s_{g'im}^-$ for $g' < g$ as well as of the

$\zeta_{g'im}$. Thus in multigroup calculations the $s_{g'im}^-$, as well as the $\zeta_{g'im}$ must be stored for each energy group, and the odd-parity source are computed recursively from the sources in the higher energy groups. In the case of upscattering, the $g'>g$ terms are taken from the previous iteration.

II. E Evaluation of Nodal Integrals

In three dimensional geometry for a fourth order approximation the vector \mathbf{f} has 35 elements, and for a quadratic approximation \mathbf{h}_γ has 6 elements. For a full P_5 approximation the nodal and interface angular trial function vectors each consist of 15 elements. The \mathbf{A} matrix is a 275,625 (35x35x15x15) element array. For hexagonal-Z geometry, with 8 interfaces per node, the \mathbf{M} matrix contains 378,000 (8x35x15x15) elements. Taking full advantage of symmetries of the coefficient submatrices, and of the orthogonality properties of the trial functions still leaves one with a staggering number of integrals to evaluate.

Performing these integrals by hand represents an intractable task. This problem is overcome through the use of a symbolic manipulation program to automate the evaluation of the cross-section independent integrals involved in generating the coefficients. To accomplish this we must break the matrices into volume and angular cross-section independent integrals which may be evaluated separately. The constituent parts are shown in Table I in dimensionless form.

The individual submatrices consist of known functions of space and angle which may be explicitly integrated. The symbolic manipulation code MATHEMATICA¹⁴ is used to evaluate the integrals. The implementation of these integrals within the symbolic manipulation code is fairly straight forward. Initially, the functional definitions of Legendre polynomials and spherical harmonics are defined within the program. These functions are then used to build up the vectors of trial functions. A simple nested do loop structure then accesses the appropriate vector elements and constructs the integrand corresponding to a particular submatrix element. The integration is then carried out over the explicitly defined domain, and the result is stored as an element in an array corresponding to a given submatrix. The array is written to an ASCII file which in turn is read by a FORTRAN program which generates a FORTRAN DATA statement containing the integrated values. The integration process is thus totally automated, and it is relatively

simple, in principle, to generate the coefficient matrices for any desired set of trial functions using Legendre polynomials, for other sets of orthogonal polynomials.

In Appendix A we show the MATHEMATICA scripts used to generate the orthogonal polynomials as well as the submatrices needed to calculate the response matrix coefficients and the flux reconstruction arrays. A three-dimensional Cartesian node is shown in Figure 1. The local nodal coordinate system appears in Figure 2. For hexagonal geometry, Figure 3 shows the orientation of the positive directions along the sides.

Table I. Integral Arrays of Spatial and Angular Basis Functions

$$\begin{aligned}
 P_{ii'}^{kl} &= \int_V dV \nabla_k f_i \nabla_l f_{i'} & H_{kl}^{mm'} &= \int d\Omega \Omega_k \Omega_l g_m^+ g_{m'}^+ \\
 U_{ii'}^l &= \int_V dV f_i \nabla_l f_{i'} & V_l^{mm'} &= \int d\Omega \Omega_l g_m^+ g_{m'}^- \\
 D_{ij\gamma} &= \int_\gamma d\Gamma f_i h_j & E_{m\gamma n} &= \int d\Omega \hat{\Omega} \cdot \hat{n}_\gamma g_m^+ k_n \\
 W_{ii'\gamma} &= \int_\gamma d\Gamma f_i f_{i'} & L_{mm'\gamma} &= \int d\Omega \hat{\Omega} \cdot \hat{n}_\gamma g_m^+ g_{m'}^-
 \end{aligned}$$

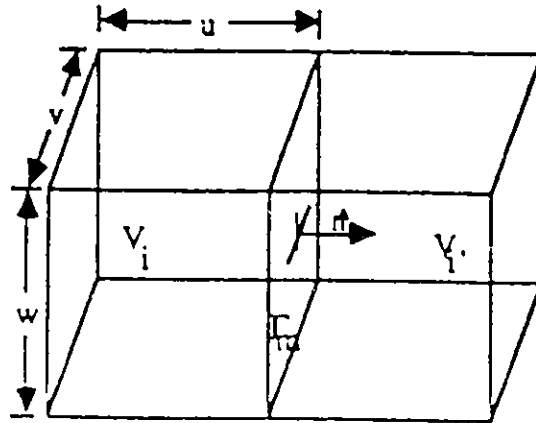


Figure 1

Three-Dimensional Cartesian Node

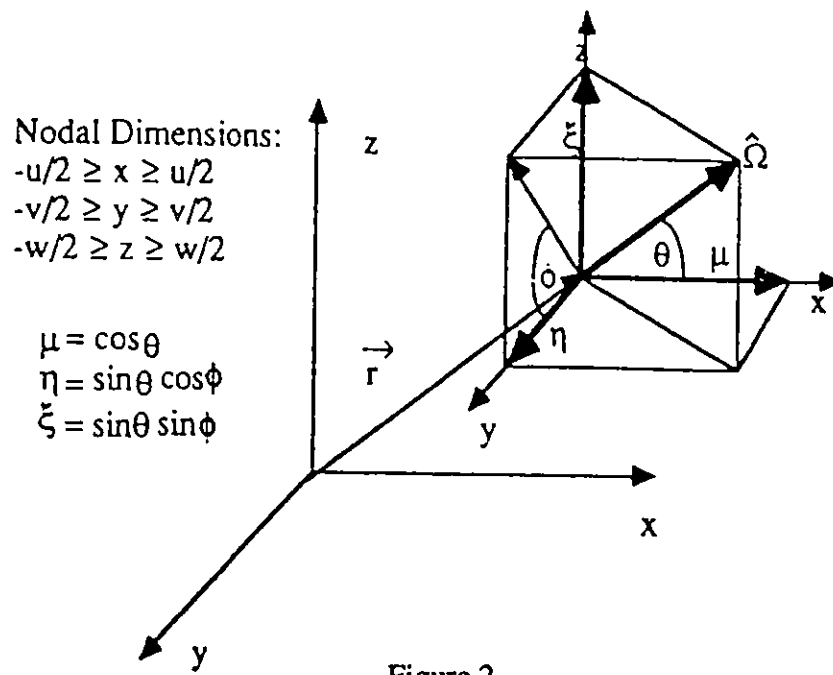


Figure 2

Local Nodal Coordinate System

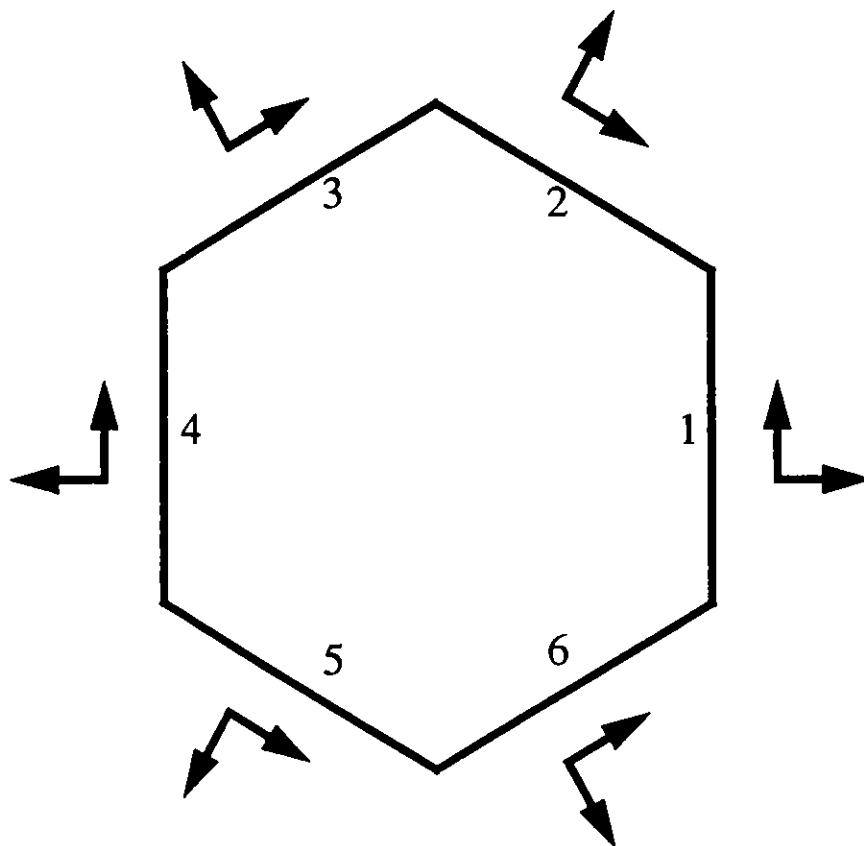


Figure 3

Orientation of the Positive Directions Along the Sides for Hexagonal Geometry

III. SOLUTION ALGORITHMS

In this Chapter we present those solution algorithms which are unique to the solution of the response matrix equations contained in VARIANT. In energy, VARIANT is a conventional multigroup code, utilizing algorithms already existing in the DIF3D shell of which it is a part. The energy-group equations are solved by fission source iteration in conjunction with a coarse mesh rebalance acceleration scheme. These iterations, referred to as outer iterations, are described in the DIF3D manual¹ and in standard texts. To perform such fission source iterations in multigroup problems, it is necessary to be able to solve for the flux moments within a group, given the group source. In VARIANT we use a red-black within-group response matrix algorithm to solve for the partial currents and then reconstruct the flux moments. In Hexagonal geometry the two-color red-black scheme must be replaced by a four color algorithm, but otherwise the logic is the same. In sections III A. we set forth the basic red-black response matrix algorithm, but for brevity the derivation of the four color algorithm is omitted. The inner iterations are accelerated by a partitioned matrix scheme similar to a synthetic diffusion acceleration method.⁶ We examine the matrix partition in Section III B. In Sections III C and III D respectively the implementation of the inner and outer iterations are presented.

III.A Red-Black Response Matrix Algorithm

The response matrix equations derived in Chapter 2 are represented in a local coordinate system centered about the node. Before we can describe the iterative solution algorithm quantitatively we must express the coupled set of response matrix equations for all of the nodes in the problem domain in terms of the local equations. We consider here the case of two-dimensional X-Y geometry, before discussing the complication of hexagonal and three-dimensional configurations.

To begin, we first divide the problem into a red-black checkerboard domain. We may then add subscripts to Eq. 2.47 or 2.101 to indicate the κ th red node

$$\mathbf{j}_{r\kappa}^+ = \mathbf{R}_{r\kappa} \mathbf{j}_{r\kappa}^- + \mathbf{q}_{r\kappa} \quad \kappa = 1, 2, 3, \dots \kappa_r \quad (3.1)$$

where in the case of anisotropic scattering both even- and odd-parity group sources are contained in $\mathbf{q}_{r\kappa}$. Suppose we now define the partial current and source vectors for the red nodes

$$\mathbf{j}_r^\pm = \begin{bmatrix} \mathbf{j}_{r1}^\pm \\ \mathbf{j}_{r2}^\pm \\ \mathbf{j}_{r3}^\pm \\ \vdots \\ \mathbf{j}_{r\kappa_r}^\pm \end{bmatrix} \quad \mathbf{q}_r = \begin{bmatrix} \mathbf{q}_{r1} \\ \mathbf{q}_{r2} \\ \mathbf{q}_{r3} \\ \vdots \\ \mathbf{q}_{r\kappa_r} \end{bmatrix} \quad (3.2)$$

and the corresponding block-diagonal global response matrices as

$$\mathbf{R}_r = \begin{bmatrix} \mathbf{R}_{r1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{r2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_{r3} \\ & & & \ddots \end{bmatrix}. \quad (3.3)$$

We may then write the global equations for the red nodes:

$$\mathbf{j}_r^+ = \mathbf{R}_r \mathbf{j}_r^- + \mathbf{q}_r. \quad (3.4)$$

The equivalent equations for the black nodes are obtained simply by replacing r subscripts with b in the foregoing procedure. The combined set of equations for red and black nodes may thus be written as

$$\begin{bmatrix} \mathbf{j}_r^+ \\ \mathbf{j}_b^+ \end{bmatrix} = \begin{bmatrix} \mathbf{R}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_b \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^- \\ \mathbf{j}_b^- \end{bmatrix} + \begin{bmatrix} \mathbf{q}_r \\ \mathbf{q}_b \end{bmatrix}. \quad (3.5)$$

We may now complete the global notation by noting that each component of the incoming partial current to a red cell is the identical to an outgoing partial current component from the adjoining black cell. This may be expressed in terms of a global connectivity matrix Π_{rb} as

$$\mathbf{j}_r^- = \Pi_{rb} \mathbf{j}_b^+ \quad (3.6)$$

for the red cells and

$$\mathbf{j}_b^- = \Pi_{br} \mathbf{j}_r^+ \quad (3.7)$$

for the black. Note that the connectivity matrices will have at most one non-zero entry per line and these will be equal to one at internal interfaces. Moreover, $\Pi_{rb}^T = \Pi_{br}$.

Equations 3.5 through 3.7 may now be used to obtain a single global response matrix equation

$$\begin{bmatrix} \mathbf{I} & -\mathbf{R}_{rb} \\ -\mathbf{R}_{br} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r \\ \mathbf{j}_b \end{bmatrix} = \begin{bmatrix} \mathbf{q}_r \\ \mathbf{q}_b \end{bmatrix}, \quad (3.8)$$

where we have defined

$$\mathbf{R}_{rb} \equiv \mathbf{R}_r \mathbf{\Pi}_{rb} \quad (3.9)$$

and

$$\mathbf{R}_{br} \equiv \mathbf{R}_b \mathbf{\Pi}_{br}, \quad (3.10)$$

which unlike \mathbf{R}_r and \mathbf{R}_b are no longer block diagonal. For brevity we have also deleted the + superscript from \mathbf{j}_r and \mathbf{j}_b since corresponding incoming partial currents have been eliminated.

The standard red-black iteration may be written as a matrix splitting in which \mathbf{R}_{br} is moved to the right side of the equation:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{R}_{br} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r \\ \mathbf{j}_b \end{bmatrix}^{l+1} = \begin{bmatrix} \mathbf{0} & \mathbf{R}_{rb} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r \\ \mathbf{j}_b \end{bmatrix}^l + \begin{bmatrix} \mathbf{q}_r \\ \mathbf{q}_b \end{bmatrix} \quad (3.11)$$

which reduces to the final two-step iteration process

$$\begin{aligned} \mathbf{j}_r^{l+1} &= \mathbf{R}_{rb} \mathbf{j}_b^l + \mathbf{q}_r \\ \mathbf{j}_b^{l+1} &= \mathbf{R}_{br} \mathbf{j}_r^{l+1} + \mathbf{q}_b, \end{aligned} \quad (3.12)$$

where l is the iteration index.

III. B The Partitioned Matrix Algorithm

Since the dimensions of transport response matrices are often large, the time per red-black iterations can be quite long. For this reason a partitioned matrix algorithm has been developed which substantially shortens the CPU time required to converge a within-group calculation.⁶ The basic idea is to partition the response matrix between a diffusion-like response matrix with only one term per interface and the larger number of higher-order space-angle interface terms required to achieve accurate transport solutions. An iteration consists of using the existing higher-order terms as a quasi-source to solve the diffusion-

like response matrix equations. Then a single sweep is made through the nodes to update the higher-order moments using the new quasi-diffusion solution. The saving in using this partitioning are often quite large. In three-dimensional P_3 calculations with bilinear spatial dependence at the interfaces, for example, there are eighteen terms per interface, meaning that the dimension of the full response matrix is eighteen times that of the quasi-diffusion calculation. In multigroup eigenvalue computations one such iteration per outer iteration is usually sufficient. In fixed source problems a larger number is required.

The procedure used in deriving the partitioned matrix algorithm parallel to considerable extent those utilized in section III. A. We first partition Eq. 2.47 or 2.101 for the κ th red node to obtain an expanded form of Eq. 3.1:

$$\begin{bmatrix} \mathbf{j}_{r\kappa}^{0+} \\ \mathbf{j}_{r\kappa}^{1+} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{r\kappa}^{00} & \mathbf{R}_{r\kappa}^{01} \\ \mathbf{R}_{r\kappa}^{10} & \mathbf{R}_{r\kappa}^{11} \end{bmatrix} \begin{bmatrix} \mathbf{j}_{r\kappa}^{0-} \\ \mathbf{j}_{r\kappa}^{1-} \end{bmatrix} + \begin{bmatrix} \mathbf{q}_{r\kappa}^0 \\ \mathbf{q}_{r\kappa}^1 \end{bmatrix}, \quad (3.13)$$

where $\mathbf{j}_{r\kappa}^{0\pm}$ represents the flat, normal partial currents for a given node, and $\mathbf{j}_{r\kappa}^{1\pm}$ represents all other higher-order current moments.

We next construct global vectors for the red nodes, analogous to Eqs. 3.2

$$\mathbf{j}_r^{0\pm} = \begin{bmatrix} \mathbf{j}_{r1}^{0\pm} \\ \mathbf{j}_{r2}^{0\pm} \\ \mathbf{j}_{r3}^{0\pm} \\ \vdots \\ \mathbf{j}_{r\kappa}^{0\pm} \end{bmatrix} \quad \mathbf{j}_r^{1\pm} = \begin{bmatrix} \mathbf{j}_{r1}^{1\pm} \\ \mathbf{j}_{r2}^{1\pm} \\ \mathbf{j}_{r3}^{1\pm} \\ \vdots \\ \mathbf{j}_{r\kappa}^{1\pm} \end{bmatrix} \quad \mathbf{q}_r^0 = \begin{bmatrix} \mathbf{q}_{r1}^0 \\ \mathbf{q}_{r2}^0 \\ \mathbf{q}_{r3}^0 \\ \vdots \\ \mathbf{q}^0 \end{bmatrix} \quad \mathbf{q}_r^1 = \begin{bmatrix} \mathbf{q}_{r1}^1 \\ \mathbf{q}_{r2}^1 \\ \mathbf{q}_{r3}^1 \\ \vdots \\ \mathbf{q}^1 \end{bmatrix} \quad (3.14)$$

The equivalent partitioned vectors may be written for each black node by substituting b for r . We may then write a partitioned matrix equation analogous to Eq. 3.5 as

$$\begin{bmatrix} \mathbf{j}_r^{0+} \\ \mathbf{j}_b^{0+} \\ \mathbf{j}_r^{1+} \\ \mathbf{j}_b^{1+} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{R}_{rb}^{00} & \mathbf{0} & -\mathbf{R}_{rb}^{01} \\ -\mathbf{R}_{br}^{00} & \mathbf{0} & -\mathbf{R}_{br}^{01} & \mathbf{0} \\ \mathbf{0} & -\mathbf{R}_{rb}^{10} & \mathbf{0} & -\mathbf{R}_{rb}^{11} \\ -\mathbf{R}_{br}^{10} & \mathbf{0} & -\mathbf{R}_{br}^{11} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^{0-} \\ \mathbf{j}_b^{0-} \\ \mathbf{j}_r^{1-} \\ \mathbf{j}_b^{1-} \end{bmatrix} + \begin{bmatrix} \mathbf{q}_r^0 \\ \mathbf{q}_b^0 \\ \mathbf{q}_r^1 \\ \mathbf{q}_b^1 \end{bmatrix} \quad (3.15)$$

To convert from local to global numbering we must make a partition of the Π_{rb} and Π_{br} matrices. These may be written as $\mathbf{j}_r^{0-} = \Pi_{rb}^0 \mathbf{j}_b^{0+}$, $\mathbf{j}_r^{1-} = \Pi_{rb}^1 \mathbf{j}_b^{1+}$, $\mathbf{j}_b^{0-} = \Pi_{br}^0 \mathbf{j}_r^{0+}$ and $\mathbf{j}_b^{1-} = \Pi_{br}^1 \mathbf{j}_r^{1+}$. Using these expressions to eliminate the incoming currents from the right of Eq. 3.8, we obtain

$$\begin{bmatrix} \mathbf{I} & -\mathbf{R}_{rb}^{00} & \mathbf{0} & -\mathbf{R}_{rb}^{01} \\ -\mathbf{R}_{br}^{00} & \mathbf{I} & -\mathbf{R}_{br}^{01} & \mathbf{0} \\ \mathbf{0} & -\mathbf{R}_{rb}^{10} & \mathbf{I} & -\mathbf{R}_{rb}^{11} \\ -\mathbf{R}_{br}^{10} & \mathbf{0} & -\mathbf{R}_{br}^{11} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^0 \\ \mathbf{j}_b^0 \\ \mathbf{j}_r^1 \\ \mathbf{j}_b^1 \end{bmatrix} = \begin{bmatrix} \mathbf{q}_r^0 \\ \mathbf{q}_b^0 \\ \mathbf{q}_r^1 \\ \mathbf{q}_b^1 \end{bmatrix} \quad (3.16)$$

where analogously to Eqs. 3.9 and 3.10 we have defined

$$\mathbf{R}_{rb}^{\alpha\beta} \equiv \mathbf{R}_r^{\alpha\beta} \Pi_{rb}^\beta \quad \alpha = 0, 1; \beta = 0, 1 \quad (3.17)$$

and

$$\mathbf{R}_{br}^{\alpha\beta} \equiv \mathbf{R}_b^{\alpha\beta} \Pi_{br}^\beta \quad \alpha = 0, 1; \beta = 0, 1 \quad (3.18)$$

We are now prepared to perform a matrix splitting. We move three of the non-zero submatrices in Eq. 3.16 to the right hand side as follows

$$\begin{bmatrix} \mathbf{I} & -\mathbf{R}_{rb}^{00} & \mathbf{0} & \mathbf{0} \\ -\mathbf{R}_{br}^{00} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{R}_{rb}^{10} & \mathbf{I} & \mathbf{0} \\ -\mathbf{R}_{br}^{10} & \mathbf{0} & -\mathbf{R}_{br}^{11} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^0 \\ \mathbf{j}_b^0 \\ \mathbf{j}_r^1 \\ \mathbf{j}_b^1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_{rb}^{01} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_{br}^{01} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_{rb}^{11} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^0 \\ \mathbf{j}_b^0 \\ \mathbf{j}_r^1 \\ \mathbf{j}_b^1 \end{bmatrix} + \begin{bmatrix} \mathbf{q}_r^0 \\ \mathbf{q}_b^0 \\ \mathbf{q}_r^1 \\ \mathbf{q}_b^1 \end{bmatrix} \quad (3.19)$$

Note that with the exception of the \mathbf{R}_{rb}^{00} submatrix the coefficient matrix on the left is lower triangular. This suggests that the iterative scheme created by adding the iteration index l to

the partial currents on the right and $l + 1$ on those to the left. If we separate the equations into two subsets, the iterative procedure is seen more clearly:

$$\begin{bmatrix} \mathbf{I} & -\mathbf{R}_{rb}^{oo} \\ -\mathbf{R}_{br}^{oo} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^o \\ \mathbf{j}_b^o \end{bmatrix}^{l+1} = \begin{bmatrix} \mathbf{q}_r^o \\ \mathbf{q}_b^o \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{R}_{rb}^{ot} \\ \mathbf{R}_{br}^{ot} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^t \\ \mathbf{j}_b^t \end{bmatrix}^l \quad (3.20)$$

and

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{R}_{br}^{tt} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^t \\ \mathbf{j}_b^t \end{bmatrix}^{l+1} = \begin{bmatrix} \mathbf{q}_r^t \\ \mathbf{q}_b^t \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{R}_{rb}^{to} \\ \mathbf{R}_{br}^{to} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^o \\ \mathbf{j}_b^o \end{bmatrix}^{l+1} + \begin{bmatrix} \mathbf{0} & \mathbf{R}_{rb}^{tt} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{j}_r^t \\ \mathbf{j}_b^t \end{bmatrix}^l \quad (3.21)$$

To solve for the $l + 1$ iterate of \mathbf{j}_r^o and \mathbf{j}_b^o we must invert the operator on the left of the first equation. But this matrix has a much smaller dimension than that in the second equation; its dimension is only equal to the number of node interfaces in the problem domain. In fact the solution is quite analogous to solving the diffusion nodal equation with only the flat components of the current at each interface. For smaller problems it may be economical to invert Eq.3.20 directly. In VARIANT we employ red-black iteration on Eq. 3.20 in the same manner described in III.B.

Once the $l + 1$ iterate of \mathbf{j}_r^o and \mathbf{j}_b^o is known, the lower triangular structure on the left of the second equation, allows the $l + 1$ iterate of \mathbf{j}_r^t and \mathbf{j}_b^t to be determined by a single successive sweep through the red and then the black nodes. For simplified coding, the red-black sweep of the second equation is replaced by a final sweep utilizing the entire response matrix. This modification may be shown to have no effect provided the quasi-diffusion calculation is converged. If the quasi-diffusion solution is not completely converged, the inclusion of \mathbf{j}_r^o and \mathbf{j}_b^o in the final red-black sweep simply gives a slight improvement in the convergence.

III. C Inner Iterations

In coding the calculation of the response matrices \mathbf{R} and \mathbf{B} of Eq. 2.47, the LINPACK and LAPACK subroutines related to matrix inversion and matrix multiplication have been used in order to improve computing times

Equation 2.67 is the basic equation solved by inner iteration once the source term is known. A guessed vector \mathbf{j}^- is used to start the calculation. Boundary conditions are used

to perform a complete sweep over the geometry and continuity conditions (outgoing partial current through an interface set equal to the incoming partial current for the adjacent node) are used to update the currents and perform inner iterations.

For boundary conditions the incoming partial currents on nodal surfaces which form part of the outer boundary of the solution domain are computed in terms of the outgoing partial current on the same surface:

$$\mathbf{j}^- = \gamma \mathbf{j}^+$$

where the albedo γ is given by:

$$\gamma = \frac{1 - 2a}{1 + 2a}$$

with “a” being the flux extrapolation constant. For the diffusion (P₁) approximation we have:

| | | |
|--|--------------|-------------------|
| zero flux boundary condition: | $a = \infty$ | $\gamma = -1$ |
| zero incoming current boundary condition | $a = 0.5$ | $\gamma = 0$ |
| zero flux at extrapolated boundaries | $a = 0.4692$ | $\gamma = 0.4692$ |
| zero net current (reflective boundary condition) | $a = 0$ | $\gamma = +1$ |

Periodic boundary conditions are treated by using the computed outgoing current across a boundary as an incoming current across the corresponding periodic boundary.

For the transport calculations, the void boundary condition is represented by Eq. 2.38. For the reflective and periodic boundary conditions, because of the spherical harmonics expansion chosen, γ is set equal to +1 or -1 according to the moment of the partial current considered.

The total number of unknowns involved in a calculation is given by the number of nodes multiplied by the number of sides of the nodes and by the total number of moments of the partial current on each side. In hexagonal-Z geometry, a full P₃ approximation, with a linear approximation on each surface results in 144 unknowns (8x6x3) per node.

The unaccelerated inner iteration is performed by ordering the nodes and carrying out a red-black (σ_1 ordering) sweep of the spatial grid. In hexagonal geometry, consistent with the existing algorithm of the DIF3D nodal option, a four color ordering has been applied.

The number of inner iterations M_g in group g for a plane is determined in a manner similar to that of the nodal option of DIF3D. Define:

$$k_g = \frac{1}{N} \sum_{n=1}^N k_g^n h \quad k_g^n = \sqrt{\sigma_g^{r,n} / D_g^n} \quad (3.22)$$

where $\sigma_g^{r,n}$ is the removal (absorption + outscattering) cross section of the node n and D_g^n is the diffusion coefficient.

The dimension "h" is taken equal to the lattice pitch in hexagonal geometry and to the square root of the area in x-y geometry. The quantity k_g is simply the averaged value of the node dimension measured in diffusion lengths. The convergence rate of the iterative procedure increases with increasing k_g since the spectral radius of the Gauss-Seidel iteration matrix decreases with increasing node size. The decreased spectral radius of the iteration matrix is due to the decreasing value of the transmission coefficient with increasing node size, which in turn increases the diagonal dominance of the global coefficient matrix. In view of this observation, plus numerical results for a number of test problems, the following simple formula is used to determine the number of inner iterations in each group g :

$$M_g = \begin{cases} 5, & k_g > 1 \\ 10, & 0.5 < k_g < 1 \\ 15, & k_g < 0.5 \end{cases}$$

The strategy adopted for the three-dimensional solution of Eq. 2.47 is consistent with the one used in the original DIF3D nodal option. Instead of considering the full matrix, \mathbf{R} is split into the plane components and the axial ones. Then the contributions coming from the incoming axial partial currents are included in the source term. The inner (plane) iterations are performed in each plane to calculate the outgoing partial currents and axial inner iterations (sweeps) are then performed with an odd-even ordering of the planes. An algorithm, not present in the original code has been introduced to calculate the number of axial inner iterations M_g^{ax} . It is similar to the planar algorithm, except that h in Eq. 3.22 is equal to the axial mesh size of the node, and the number of iterations is now selected to be

$$M_g^{\text{ax}} = \begin{cases} 2, & k_g > 0.5 \\ 4, & k_g < 0.5 \end{cases}$$

In coding the algorithms, much care has been taken in order to insure a maximum use of vectorization, especially in solving Eqs. 3.20 and 3.21, which are fully and optimally vectorized,

Several attempts have been made to take advantage of symmetry of the matrix and the presence of a significant number of zero values in the response matrix due to the orthogonality of the expansion functions. Unfortunately, the penalty associated with the use of an indirect addressing and a partial loss of vectorization has discouraged such an approach. Therefore, the full response matrices are presently used for the computation of the partial currents.

III. D Outer Iterations

Once the partial currents are calculated the average flux in the node may be evaluated using Eq. 2.52. When the scalar flux moments are known, the K_{eff} calculation can proceed as in a standard code, by the evaluation of the fission source. Then, a new outer iteration can be performed with the evaluation of the inner iteration process of the required quantities group by group (currents, fluxes, scattering sources).

An attempt was made to accelerate the convergence of the outer iterations by introducing the Chebychev polynomial method. The already existing machinery used by the finite difference option of DIF3D was employed. Unfortunately, the acceleration method turned out to be ineffective, and sometimes slowed convergence. No clear reasons have been found to explain such a behavior. The Chebychev polynomial acceleration is independent of the algorithm used to solve the fixed source problem related to a single outer iteration, provided that the inner iterations are sufficiently converged. Nevertheless, it was found that greatly increasing the number of full-matrix inner iterations has no positive impact on the efficiency of the Chebychev method when applied to the variational nodal method. Moreover, a similar trend has been observed when this acceleration method has been applied to the original DIF3D nodal option. A possible explanation is related to the presence of flux moments, and therefore of fission source moments. In this situation, the dominance ratio calculated to evaluate the acceleration parameters will not be representative of the entire iterative matrix because only the first moment of the fission source is used in its determination.

For this reason it was decided to apply and adapt the algorithms already existing in the DIF3D nodal option to accelerate the outer iterations: coarse mesh rebalancing and

asymptotic source extrapolation. Full details of these methods are given in Ref. 15. We only note that in calculating the leakage term of the coarse mesh rebalance equation we use only the first moment of the partial currents because of the physical meaning of the quantity. Slight differences are also caused by the fact that in our algorithm the partial currents appear as integrated quantities over the node surface, whereas in the original DIF3D nodal option, the partial currents are averaged values. The efficiency of the coarse mesh rebalance acceleration and the asymptotic source extrapolation methods has been verified for use in conjunction with the variational nodal method by several numerical tests.

IV. NUMERICAL CALCULATIONS

Several numerical calculations have been carried out in order to determine the optimum order of spatial approximation for the flux, source and leakage dependence. Based on these results, it was decided to adopt a fourth order expansion for the flux dependence inside the node. A linear leakage spatial dependence has been adopted together with a quadratic expansion for the source. When using simplified spherical harmonics, a flat approximation on the leakage term is used. Because of error compensation, this approximation was found to give better results. Of course, all these approximation have been left parametrized in the code in such a way that the user can change them.

IV. A Two Dimensional Results

In Table II we show the results obtained from an x-y model of the EBR-II reactor similar to the one defined in reference 24. The model has been modified in order to enhance the transport effect (more than 3% of $\Delta K/K$). A nine group energy structure is used.

Table II. EBR-II x-y Geometry (enhanced transport effect)

| Type of Calculation | K_{eff} | CPU Time (sec) ^a |
|---------------------------|-----------|-----------------------------|
| $S_4 (\Delta)$ | 0.99314 | 3.5 |
| $S_8 (\Delta)$ | 0.99070 | 5.5 |
| $S_8 (4\Delta)$ | 1.01417 | 24.4 |
| $S_8 (16\Delta)$ | 1.01980 | 20.0 |
| VARIANT P_1 (diffusion) | 0.99084 | 0.9 (0.3 + 0.6) |
| VARIANT P_{31} | 1.02361 | 2.1 (1.1 + 0.6) |
| VARIANT Simpl. P_3 | 1.02409 | 1.2 (0.6 + 0.6) |
| VARIANT P_3 | 1.02207 | 6.8 (4.2 + 2.6) |
| Reference | 1.02199 | - |

^aFor VARIANT the breakdown of, respectively, response matrix coefficients and outer iteration CPU calculation time is given in parenthesis.

For this case we report also the S_N results for different angular approximations and different mesh sizes (the basic mesh grid Δ , used in the nodal calculation, has been successively refined by dividing the mesh size by two (4Δ) and four (16Δ)). The S_N calculations have been carried out using the highly optimized TWODANT code²⁵. The reference solutions have been obtained by extrapolation of the refined TWODANT finite difference calculations.

The S_N calculations are very poor for the basic mesh grid where both S_4 and S_8 give results comparable with the diffusion calculation (P_1 solution of VARIANT). The S_8 (16Δ) calculation, which is still not as accurate as the P_3 solution of VARIANT, requires a factor 3 more computation time. All calculations, as well as the ones presented in the following, have been carried out on a RISC 6000/350 IBM workstation.

The P_{31} (corresponding to the reduced angular approximation) and the simplified spherical harmonic calculations provide comparable results. They overestimate the reference solution by less than 0.2% of $\Delta K/K$, and, therefore, account for more than 90% of the total transport effect. The simplified spherical harmonic approximation requires almost half the time of the P_{31} calculation.

We also observe that for the full P_3 approximation most of the time is spent in the coupling coefficient calculation. This time is 14 times larger than that required for the corresponding P_1 (diffusion) calculation. The reason is related to the different number of floating point operation, which increases as the square of the response matrix dimension.

In Table III we show results for an EBR-II hexagonal 2D model. In this case the total transport effect is of the order of 1.3% $\Delta K/K$. The S_N calculations were carried out with the TWOHEX²⁶ code, using 6 triangles (Δ) or 24 triangles (4Δ) per subassembly. Recall that in the case of the variational nodal method, only one node per subassembly is used.

Comparing computing times, we can note that the TWOHEX S_4 (Δ) calculation requires 5 times more CPU time than the VARIANT P_3 solution, which is in satisfactory

Table III. EBR-II Hexagonal 2-D Geometry

| Type of Calculation | K_{eff} | CPU Time (sec) ^a |
|---------------------------|-----------|-----------------------------|
| $S_4 (\Delta)$ | 1.03240 | 89.5 |
| $S_8 (\Delta)$ | 1.03274 | 268.2 |
| $S_8 (4\Delta)$ | 1.03285 | 1053 |
| VARIANT P_1 (diffusion) | 1.01882 | 2.6 (0.4 + 2.2) |
| VARIANT P_{31} | 1.03641 | 6.0 (3.7 + 2.3) |
| VARIANT Simpl. P_3 | 1.03396 | 1.8 (0.8 + 1.0) |
| VARIANT P_3 | 1.03326 | 17.6 (9.0 + 8.6) |
| Reference | 1.03289 | - |

^aFor VARIANT the breakdown of, respectively, response matrix coefficients and outer iteration CPU calculation time is given in parenthesis.

agreement with the reference solution. Again a significant time is spent in computing the response matrix coefficients. This is due to the presence of a large number (19) of nodes with different compositions. The simplified spherical harmonic calculation gives a solution that is clearly more accurate than the P_{31} solution and requires more than a factor 3 less time.

We also point out that, in hexagonal geometry, VARIANT provides better results than the nodal diffusion approximation of DIF3D¹⁵ for diffusion calculations when the solutions from both methodologies are compared against a reference solution obtained by extrapolating the finite difference results to a zero mesh size²⁷. This is related to the fact that the variational nodal method employs a complete polynomial expansion to describe the flux intranodal spatial dependence, whereas in the case of the DIF3D nodal option, cross terms are neglected. This is not the case in Cartesian geometry where the two methods give almost identical results for diffusion calculation when similar approximations are used for the flux, source and leakage spatial dependence.

IV. B Three Dimensional Results

The Takeda benchmark²⁸ model 2 has been used to test the performance of VARIANT in three dimensional Cartesian geometry. This model is representative of a small FBR reactor and employs a four group energy structure. Results are shown in Table IV. The reference solution is provided by the ANL VIM²⁹ Monte Carlo code using the same multigroup cross section set. Table IV also exhibits results (from reference 28) of S_N solutions calculated by the THREEDANT code²⁵.

Table IV. Takeda Benchmark Model 2 x-y-z Geometry Small FBR

| Type of Calculation | K_{eff} Rod Out | K_{eff} Rod In | Control Rod Worth | CPU Time (sec) ^a |
|------------------------------|---------------------|---------------------|--------------------|-----------------------------|
| Monte Carlo VIM | 0.97344± 0.00036 | 0.95988± 0.00038 | 0.1451± 0.00057 | ~8 hrs |
| Threedant S_8 | 0.97348 | 0.95931 | 0.1517 | - |
| DIF3D Nodal Transport Option | 0.97138 | 0.95701 | 0.1546 | 11.7 (0.5 + 11.2) |
| VARIANT P_1 | 0.96913 | 0.95430 | 0.1604 | 44.5 (0.3 + 44.2) |
| VARIANT P_{31} | 0.97228 | 0.95814 | 0.1518 | 46.9 (1.6 + 45.3) |
| VARIANT Simpl. P_3 | 0.97429 | 0.96028 | 0.1497 | 28.7 (0.6 + 18.1) |
| VARIANT P_3 | 0.97349 | 0.95942 | 0.1506 | 562 (25 + 537) |

^aFor VARIANT the breakdown of, respectively, response matrix coefficients and outer iteration CPU calculation time is given in parenthesis.

The P_3 VARIANT results are very similar to the S_8 calculations. Both solutions are in very good agreement with the reference Monte Carlo calculation. Of course, the CPU time required by VARIANT is more than one order of magnitude less than that needed by VIM.

The simplified spherical harmonic solution is again more accurate than the P_{31} solution and requires even less CPU time than the diffusion solution. For comparison, we have also displayed the results obtained by the transport option of the DIF3D nodal

calculation. Although the time required is less than that for the simplified spherical harmonics, we observe that the latter solution provides better results for both K_{eff} and control rod worth.

Finally to test the code in 3D hexagonal geometry, we have considered the simplified model of EBR-II provided in reference 24 (see figure 4). Results are provided in Table V.

Table V. EBR-II Hexagonal-z Geometry

| Type of Calculation | K_{eff} | CPU Time (sec) ^a |
|----------------------|-------------------|-----------------------------|
| Monte Carlo VIM | 1.20423 ± 0.00045 | ≈ 4 days |
| VARIANT P_1 | 1.17268 | 73 (2 + 71) |
| VARIANT P_{31} | 1.19523 | 213 (42 + 171) |
| VARIANT Simpl. P_3 | 1.20292 | 47 (5 + 42) |
| VARIANT P_3 | 1.20349 | 1542 (308 + 1234) |

^aFor VARIANT the breakdown of, respectively, response matrix coefficients and outer iteration CPU calculation time is given in parenthesis.

The P_3 variational nodal eigenvalue is within a few tenths of one percent of the VIM Monte Carlo code result. We note that the VIM calculation required a few days of CPU time against the 26 minutes required by VARIANT.

The simplified spherical harmonics calculation provides far superior results to the P_{31} solution (96% of the total transport effect against 72%) with a CPU time that again is lower than the diffusion calculation. This is mainly due to the lower number of outer iterations required to converge. The number of unknowns are the same for the diffusion and the simplified harmonic approximation (16 per node).

In Table VI the comparison for the hexagonal row axially integrated power obtained is shown. As we can see, the discrepancy between the P_3 and the VIM values does not exceed 0.5%. The diffusion solution has discrepancies of the order of 5% in the reflector and blanket regions, whereas the maximum error for the simplified spherical harmonic calculation is of the order of 1% and occurs in the first ring of the reflector region.

Figure 4

EBR-II Representation for a Three Dimensional Criticality Problem

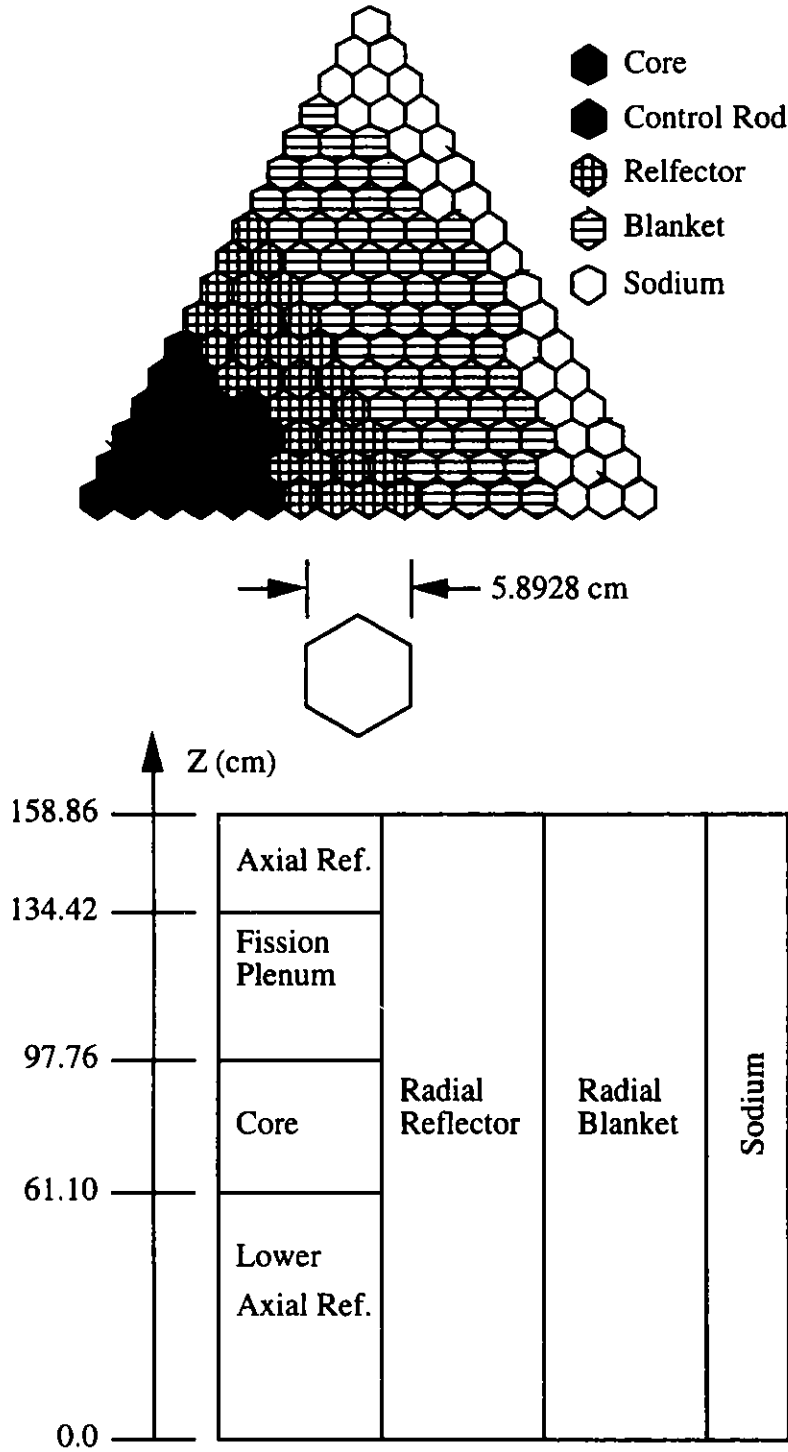


Table VI. EBR-II Ring Axially Integrated Power (MWth)

| Ring | VIM | VARIANT P ₁ ^a | VARIANT Simpl. P ₃ ^a | VARIANT P ₃ ^a |
|------|---------------------------|-------------------------------------|--|-------------------------------------|
| 1 | 3.440 | 3.395 (-1.31) | 3.425 (-0.44) | 3.441 (0.03) |
| 2 | 3.385 | 3.343 (-1.25) | 3.373 (-0.35) | 3.387 (0.06) |
| 3 | 6.500 | 6.431 (-1.06) | 6.486 (-0.22) | 6.508 (0.12) |
| 4 | 9.084 | 9.027 (-0.63) | 9.098 (0.15) | 9.117 (0.36) |
| 5 | 10.975 | 10.938 (-0.34) | 11.009 (0.31) | 11.010 (0.32) |
| 6 | 12.089 | 12.065 (-0.20) | 12.089 (0.00) | 12.067 (-0.78) |
| 7 | 12.457 | 12.606 (1.20) | 12.432 (-0.20) | 12.400 (-0.46) |
| 8 | 14.435 × 10 ⁻² | 15.166 × 10 ⁻² (5.06) | 14.587 × 10 ⁻² (1.05) | 14.492 × 10 ⁻² (-0.39) |
| 9 | 14.65 × 10 ⁻² | 15.40 × 10 ⁻² (5.17) | 14.763 × 10 ⁻² (0.76) | 14.676 × 10 ⁻² (0.17) |
| 10 | 13.112 × 10 ⁻² | 13.816 × 10 ⁻² (5.37) | 13.222 × 10 ⁻² (0.84) | 13.143 × 10 ⁻² (0.24) |
| 11 | 95.30 × 10 ⁻³ | 10.058 × 10 ⁻² (5.53) | 96.098 × 10 ⁻³ (0.83) | 95.595 × 10 ⁻³ (0.30) |
| 12 | 64.846 × 10 ⁻² | 68.179 × 10 ⁻² (5.14) | 65.316 × 10 ⁻² (0.72) | 65.158 × 10 ⁻² (0.48) |
| 13 | 37.505 × 10 ⁻² | 39.165 × 10 ⁻² (4.43) | 37.627 × 10 ⁻² (0.33) | 37.614 × 10 ⁻² (0.29) |
| 14 | 21.972 × 10 ⁻² | 22.829 × 10 ⁻² (3.90) | 21.960 × 10 ⁻² (-0.05) | 21.984 × 10 ⁻² (0.05) |

^aDiscrepancy (in percent) with respect to the reference VIM value is given in parenthesis.

V. USER INFORMATION

VARIANT can be executed as an option in a standard DIF3D calculation. This section highlights information of particular interest to users of the variational nodal option and supplements the documentation provided in references 1 and 15 and the description of the BCD input files.

V. A Data Management

In order to minimize differences with the original coding of DIF3D, data management has been kept with the same strategy of two large blocks of work space (fast and extended core memory) even as two-level computers are becoming obsolete. The philosophy of containment stays the same: extended core memory contains arrays which can be stored on external data files.

Because a large memory size is required to store nodal coupling coefficients, a special strategy has been introduced for their management. First, all the matrices involved in the response matrix equation and in the flux and source evaluations are mapped for unique non-zero values. This mapping is done before the total memory requirement is evaluated and demands a sizable quantity of memory (of the order of the one normally required in the fast core memory array).

If the memory allocated by the user for the extended core array is sufficient, matrices are used as they are; otherwise they are compressed and only unique elements are stored along with their location in the original matrices. Of course, with the compressed matrices computation time during outer iterations is penalized because of the use of indirect addressing.

Due to the methodology of modeling anisotropic scattering in VARIANT, a new COMPXS file structure has been implemented. The modification has been made in order to

preserve compatibility with already existing codes that use COMPXS files. The description of the new file structure is presented in Appendix B.

To enhance the performance on vector machines a new ordering of the partial current moments has been adopted. This is reflected in the new specification of the NHFLUX file structure shown in Appendix C. Contrary to the COMPXS file, the NHFLUX generated by VARIANT will be incompatible with existing codes that use partial currents with moments greater than first.

Recall that in order to reduce storage and computation time, the flux in VARIANT is evaluated, using Eq. 2.52 -- only up to the number of moments of the source expansion. This is done because only these moments of the flux are needed to compute the new outer iteration source distribution. Therefore, only these moments are stored on the NHFLUX file. In the case of the anisotropic calculation, the even-parity angular flux moments are also evaluated and stored on NHFLUX, because they are needed in the anisotropic source computation.

A post processor program, that reads the flux moments from the NHFLUX file, reconstructs the flux locally at designated points of specified subassemblies and computes the related reaction rates, will be soon made available.

V. B Variational Nodal Parameters

A new card 12 of the BCD input file A.DIF3D (see Appendix D) has been introduced to specify the nodal variational parameters. Some comments on their meaning and use follow.

V. B.1 Nodal Spatial Approximation

Default values for within the node flux spatial approximations are fourth order in Cartesian geometry and sixth order in hexagonal geometry. Linear dependence of the leakage

on the surface of the node is also recommended. These values eliminate rank deficiency from the nodal coupling coefficient matrices.²²

Figure 5 shows the required internal order for a given surface approximation in order to insure rank sufficiency. For a linear approximation of the spatial leakage slope, a third order intra nodal flux expansion is needed for Cartesian geometry, and a sixth order expansion is needed for hexagonal geometry. For a very tight convergence criteria (e.g., $\sim 10^{-7}$), rank deficiency will result in a lack of convergence. For relaxed convergence criteria (e.g., $\sim 10^{-5}$), third order for Cartesian and fourth order for hexagonal geometry can be safely used in connection with linear approximation on the leakage term. For three-dimensional hexagonal geometries, the axial expansion is kept to fourth order in order to minimize the size of the response matrices while insuring rank sufficiency.

The order of the default source expansion polynomial is taken equal to one greater than the surface approximation. It has been found that in some cases (especially for thermal reactor configurations) an order equal to that used for the intra-nodal flux is necessary to insure good power distribution results.

V. B.2 Angular Approximations

Specification of the P_1 approximation for both flux and leakage will provide diffusion results. Using P_3 for the flux and P_1 for the leakage will trigger the use of the reduced angular approximation. With a negative value for the angular approximation variable input, the code will use the corresponding simplified spherical harmonics. Because of error compensation, the best results in this case are most often obtained with a flat (0) approximation for the spatial dependence of the leakage on the surface of the node. The flux angular expansion cannot be lower than the anisotropic order NPNO specified later.

X-Y

Surface Expansion


| | | | | |
|--------------------|-------|------|------|-------|
| | | 0(4) | 1(8) | 2(12) |
| Internal Expansion | 1(3) | 3 | 3 | 3 |
| | 2(6) | 4 | 5 | 6 |
| | 3(10) | 4 | 7 | 10 |
| | 4(15) | 4 | 8 | 12 |


Hexagonal

Surface Expansion

| | | | | |
|--------------------|-------|------|-------|-------|
| | | 0(6) | 1(12) | 2(18) |
| Internal Expansion | 1(3) | 3 | 3 | 3 |
| | 2(6) | 5 | 5 | 6 |
| | 3(10) | 6 | 9 | 10 |
| | 4(15) | 6 | 11 | 14 |
| | 5(21) | 6 | 11 | 17 |
| | 6(28) | 6 | 12 | 18 |

$n(m) \equiv$ expansion order (number of terms)

 \equiv dimension deficient rank

 \equiv projection deficient rank

 \equiv sufficient rank

Figure 5

M Matrix Ranks for Two-Dimensional Geometries

V. B.3 Asymptotic Extrapolation Sentinel

A new option, (asymptotic extrapolation) has been introduced. This option is invoked by setting the sentinel to -1. In this case only the fission sources are accelerated and no extra space is needed to store previous outer iteration partial currents. No significant penalty has been observed in the performance of the acceleration using this option.

V. B.4 Anisotropic Scattering Order (NPNO) and Extended Transport Approximation (NXTR)

No anisotropic order greater than either MAXORD (scattering order of cross sections on the COMPXS file) or the flux angular expansion is allowed.

An option (NEXTR parameter) to invoke the use of the total cross section, transport cross section, or extended transport approximation is described below.

NEXTR set to a negative value is intended solely to perform comparison calculation and should not be used for any other purpose. This setting forces the use of the zero moment of the total cross section for isotropic calculations, and the use of the transport cross section for anisotropic calculations.

The default value of 0 is strongly recommended for NEXTR. With this value the transport cross section is used for isotropic calculation. For the anisotropic calculations the total cross section is used unless the value of NPNO is lower than MAXORD. In this case the BHS approximation³⁰ is applied (the extended transport approximation corrects the total cross section by taking into account the NPNO+1 order of the anisotropic scattering).

If NEXTR is specified to set at a value N greater than NPNO, an extended transport approximation is applied from NPNO+1 to NXTR. Be aware that if NXTR is greater than NPNO+1, this correction is done at the risk of the user; there is no proof that such correction will give reasonable results.

With these premises, having a pair of identical values for NPNO and NXTR (00,11,22,33, etc.) will be strictly equivalent to having: 00,10,20,30, etc.

V. B.5 Nodal Coupling Coefficient Packing Option

The default value (0) results in no packing of the nodal coupling matrices unless the array length provided by the user for the extended core memory is not sufficient. The user can force the packing by providing a value of 1 for this option. This sometimes can be useful when it will allow the problem to run with all the group constants (cross sections, fluxes, currents) in core, thus reducing the input/output operations and therefore compensating for the increase in CPU time resulting from the use of indirect addressing to unpack the matrices during the iterations calculations. Workstations with poor input/output performances seem to benefit most from this strategy.

V. B.6 Radial Inner Iteration Algorithm

The default value (0) implies the use of the partitioned algorithm. For an outer iteration, n inner iteration are first performed on the first moment of the partial currents, the higher moments contributions are included in the source term. This is followed by a full sweep on all the moments. Sometimes, the full sweep matrix algorithm is necessary to avoid convergence problems. This is performed by applying the partitioned algorithm cycle n times, where n is the total number of inner iterations.

When the maximum number of outer iterations or convergence is reached, proper convergence of the inner iterations is checked. The last outer iteration is performed using the full sweep matrix algorithm. The inner iteration convergence criteria is identical to the pointwise fission source specified in convergence criteria card 5. Only the first moment of the partial currents is checked.

V. C Limitations

Enough memory must be allocated to contain all the information for at least one energy group. Flux and source expansions up to sixth order are allowed in hexagonal geometry, fourth order in Cartesian geometry. Partial current expansion up to second order are allowed. Angular and scattering expansion of up to P_5 are allowed. For highly heterogeneous reactor configurations involving thousands of different node types, calculation and storage of response matrices represents the primary computational cost. In problems of this type, it is highly desirable to store as many response matrices as possible in fast memory.

V. D Programming Information

The programming structure of the nodal option of DIF3D has been retained for the VARIANT option. Many of the existing subroutines have been modified, keeping essentially the same functionality, and added with a new name where a V replaces an N. The list of modified subroutines can be found in Table VII. Table VIII lists the names of the new subroutines that were not part of the original version of DIF3D. The call tree for the main branches of the VARIANT option is shown in Fig. 6, keeping in mind that, referring to Fig. 7.1 of Ref. 15, VHINIT, VHSST and VSINIT have replaced NHINIT, NHSST and XSINIT. Subroutines starting with D belong to the LAPACK and LINPACK mathematical package. A new common block /VARIAN/, which contains parameters specific to the nodal variational option, has been added. Finally in Table IX we show the list of the original DIF3D subroutines that have been modified, without changing their name, in order to accommodate the new VARIANT option and the new COMPXS file structure with anisotropic scattering capability.

ACKNOWLEDGMENTS

The authors would like to acknowledge the contributions of K. L. Derstine, P. Finck, H. Kahlil, and R. Jacqmin in providing frequent consultation and reference solutions. The availability of the algorithms developed by R. Lawrence for the DIF3D nodal option greatly facilitated the implementation of VARIANT. This work was performed under the U.S. Department of Energy contract WQ-37-109-ENG-38 at Argonne National Laboratory.

**Table VII. List of Modified Original DIF3D Subroutines,
With Name Changes where a V Replaces an N**

| | | | |
|--------|--------|--------|--------|
| RVHFLX | VHCCPT | VLXZ | VSERRN |
| VALINT | VHCMPT | VMBKRG | VSGET1 |
| VBLADD | VHCORE | VMFSYM | VSGET2 |
| VCCEL | VHDISK | VMINIT | VSINIT |
| VCCL3D | VHEDDM | VMJBDJ | VSTOU1 |
| VCHEx | VHGEOM | VMMTRX | VSTOU2 |
| VCHExB | VHINED | VNHCCC | VSUPDT |
| VCMPXS | VHINIT | VNHFIN | VUTR1 |
| VCZ | VHINNR | VNHOUT | VUTR2 |
| VCZB | VHOED0 | VNHSTT | VUTR3 |
| VDSCTM | VHPEAK | VONVCK | VUTR4 |
| VEXBAL | VHPKED | VRCFIS | VUTR5 |
| VEXREA | VHPNT | VRCHEX | VXINIT |
| VFSINI | VHSHAP | VRCSCT | VXSHAP |
| VFXREA | VHSST | VRCZI | VXYZCC |
| VHCC2D | VHXSEC | VSCREV | WVHFLX |
| VHCC3D | VLXHEX | VSEdit | |

**Table VIII. List of the New Subroutines that Were Not
Part of the Original Version of DIF3D**

| | | | |
|--------|--------|--------|--------|
| CONCKI | MACXY | SRCSCP | VCOH3D |
| HALCOP | MACXYZ | TVACBC | VCOXY |
| MACH2D | NPKCC | UNIEL | VCOXYZ |
| MACH3D | PCXY | VCOH2D | ZERMAP |

Table IX. List of Modified Original DIF3D Subroutines Without Name Changes

| | | | |
|--------|--------|--------|--------|
| BCDINP | EDITCR | SSTATE | OVL2 |
| BININP | LINKR1 | COPIER | OVL3 |
| DIF3D | LINKR2 | DOMODS | OVL5 |
| DSST01 | NHSIGA | FARSET | SVSCAT |
| DSST02 | PDIF3D | HMG4C | UPDATE |
| DSST03 | RADF3D | ISOR58 | WREC1 |
| DXSREV | SSINT | MAXBND | WREC4 |

Figure 6. Call Tree for the Main Branches of the VARIANT Option

```

VHINIT  CLOSCF
.        DEFICF
.        HEXMAP
.        ICRED
.        ICRIT
.        NHZMAP
.        OPENCF
.        PURGCF
.        PURGE
.        PUTM
.        VHCCEPT  ICRED
.        VHCNPT
.        VHCORE    DEFICF
.                DELECF
.                EDITCR
.                ERROR
.                INTSET
.                LINES
.                PURGCF
.                PURGE
.                WIPOUT
.        VHDISK   DEFIDF
.                DOPC
.                ERROR
.        VHGION   GETIJ
.                NPKCC
.                ERROR
.                LINES
.                MACH2D  DDCOPY
.                .      DGEMM
.                .      DGETF2
.                .      DGETRI
.                .      DPOTF2
.                .      DPOTRI
.                .      DSCAL
.                .      DSYMM
.                .      HALCOP
.                .      MACH3D  DGEMM
.                .      DGETF2
.                .      DGETRI
.                .      DPOTF2
.                .      DPOTRI
.                .      DSCAL
.                .      DSYMM
.                .      HALCOP
.                .      MACXY   DDCOPY
.                .      DGEMM
.                .      DGETF2
.                .      DGETRI
.                .      DPOTF2
.                .      DPOTRI
.                .      DSCAL
.                .      DSYMM
.                .      HALCOP
.                .      MACXYZ  DDCOPY
.                .      DGEMM
.                .      DGETF2
.                .      DGETRI
.                .      DPOTF2
.                .      DPOTRI
.                .      DSCAL

```

Figure 6. Call Tree for the Main Branches of the VARIANT Option (Cont'd.)

| | | | | | |
|-------|--------|--------|--------|--------|--------|
| . | . | . | . | DSYMM | |
| . | . | . | . | HALCOP | |
| . | . | . | . | PUTM | |
| . | . | . | . | UNIEL | |
| . | . | . | . | VCOH2D | |
| . | . | . | . | VCOH3D | |
| . | . | . | . | VCOXY | |
| . | . | . | . | VCOXYZ | |
| . | . | . | . | ZERMAP | |
| . | VHINED | LINE | | | |
| . | VHPNT | INTSET | | | |
| . | . | PUTM | | | |
| . | WIPOUT | | | | |
| VHSST | CLOSCF | | | | |
| . | CLODF | | | | |
| . | ERROR | | | | |
| . | FLTSET | | | | |
| . | LINE | | | | |
| . | NHOED0 | | | | |
| . | OPENCF | | | | |
| . | OPENDF | | | | |
| . | PURGCF | | | | |
| . | PURGE | | | | |
| . | PUTM | | | | |
| . | VNHCC | BLKGET | | | |
| . | . | BLKPUT | | | |
| . | . | FINGET | | | |
| . | . | FINPUT | | | |
| . | . | ICRED | | | |
| . | . | ICRIT | | | |
| . | . | NODVOL | | | |
| . | . | PCRED | | | |
| . | . | PCRIT | | | |
| . | . | VHCC2D | FILGAM | | |
| . | . | . | GETBND | | |
| . | . | . | MACH2D | ... | etc... |
| . | . | VHCC3D | FILGAM | | |
| . | . | . | GETBND | | |
| . | . | . | MACH3D | ... | etc... |
| . | . | VHINN | | | |
| . | . | VHXSEC | FLTSET | | |
| . | . | VXYCC | FILGAM | | |
| . | . | . | GETBND | | |
| . | . | . | MACHY | ... | etc... |
| . | . | VXYZCC | FILGAM | | |
| . | . | . | GETBND | | |
| . | . | . | MACHYZ | ... | etc... |
| . | VNHFIN | BLKGET | | | |
| . | . | BLKPUT | | | |
| . | . | FINGET | | | |
| . | . | FINPUT | | | |
| . | . | FLTSET | | | |
| . | . | ICRED | | | |
| . | . | NODVOL | | | |
| . | . | OPENCF | | | |
| . | . | OPENDF | | | |
| . | . | PCRED | | | |
| . | . | PCRIT | | | |
| . | . | PURGE | | | |

Figure 6. Call Tree for the Main Branches of the VARIANT Option (Cont'd.)

```

.      .      VHEDDM      CLOSCF
.      .      .          CPYFIL
.      .      .          DEFICF
.      .      .          OPENCF
.      .      .          PURGCF
.      .      VHFSYM      TVACBC
.      .      VHXSEC      ...etc...
.      .      VXSHAP
.      .      WDIF3D
.      .      WIPOUT
.      .      XREAD
.      .      VNHOUT     BLKGET
.      .      .          BLKPUT
.      .      .          FINGET
.      .      .          FINPUT
.      .      .          FLTSET
.      .      .          OPENCF
.      .      .          VHOEDT     VHOEDO      ERROR
.      .      .          .          LINEX
.      .      VUTR1      ICRED
.      .      .          PCRED
.      .      .          PCRIT
.      .      .          VCCEL      FEQUAT
.      .      .          .          VCHEXB      TVACBC
.      .      .          VCCL3D     FEQUAT
.      .      .          .          VCZB      TVACBC
.      .      VUTR2      BLKGET
.      .      .          FINGET
.      .      .          FLTSET
.      .      .          ICRED
.      .      .          PCRED
.      .      .          PCRIT
.      .      .          SRCSCF
.      .      .          VRCFIS
.      .      .          VRCSCT
.      .      VUTR3      ICRED
.      .      .          PCRED
.      .      .          PCRIT
.      .      .          PCKY
.      .      .          VCHEX      CONCKI
.      .      .          .          DCOFY
.      .      .          .          PCOPY
.      .      .          .          VCHEXB      ...etc...
.      .      .          VCZ        VCZB      ...etc...
.      .      .          VRCHEX     DCOFY
.      .      .          .          VCZB      ...etc...
.      .      .          VRCZ1
.      .      VUTR4      FLTSET
.      .      .          ICRED
.      .      .          PCRED
.      .      .          PCRIT
.      .      .          VLXHEX
.      .      .          VLXZ
.      .      .          VMMTRX     VMBKRG
.      .      .          .          VMFSYM      TVACBC
.      .      .          .          VMJBDY
.      .      .          VSUPDT
.      .      VUTR5      CMSOLV
.      .      .          ICRED
.      .      .          PCRED
.      .      .          PCRIT
.      .      .          TIMER

```

Figure 6. Call Tree for the Main Branches of the VARIANT Option (Cont'd.)

| | | | | |
|--------|--------|--------|--------|-------|
| . | . | . | VONVCK | TIMER |
| . | . | . | VSERRN | |
| . | VNHSTT | BLKGET | | |
| . | . | BLKPUT | | |
| . | . | CLOSCF | | |
| . | . | FINGET | | |
| . | . | FINPUT | | |
| . | . | FLTSET | | |
| . | . | ICRED | | |
| . | . | OPENCF | | |
| . | . | PCRED | | |
| . | . | PCRIT | | |
| . | . | SEEK | | |
| . | . | VEKREA | ERROR | |
| . | . | . | FLTSET | |
| . | . | . | LINES | |
| . | . | . | PCRED | |
| . | . | . | PCRIT | |
| . | . | . | REED | |
| . | . | . | SEEK | |
| . | . | VFSINI | | |
| . | . | VFXREA | ERROR | |
| . | . | . | INTSET | |
| . | . | . | LINES | |
| . | . | . | PCRED | |
| . | . | . | PCRIT | |
| . | . | . | REED | |
| . | . | VMINIT | FLTSET | |
| . | . | VKINIT | FLTSET | |
| . | WIPOUT | | | |
| | | | | |
| VSINIT | CLOSCF | | | |
| . | ERROR | | | |
| . | LINES | | | |
| . | OPENCF | | | |
| . | PNTGET | | | |
| . | PURGCF | | | |
| . | PURGE | | | |
| . | PUTM | | | |
| . | SEEK | | | |
| . | VSGET1 | BLKPUT | | |
| . | . | CLODF | | |
| . | . | ERROR | | |
| . | . | FEQUAT | | |
| . | . | FINPUT | | |
| . | . | FLTSET | | |
| . | . | IEQUAT | | |
| . | . | LINES | | |
| . | . | OPENDF | | |
| . | . | REED | | |
| . | . | RITE | | |
| . | . | VSEDIT | CHRFLT | |
| . | . | . | LINES | |
| . | VSGET2 | BLKPUT | | |
| . | . | CLODF | | |
| . | . | FINPUT | | |
| . | . | OPENDF | | |
| . | . | REED | | |
| . | WIPOUT | | | |

REFERENCES

1. K. L. Derstine, "DIF3D: A Code to Solve One-, Two- and Three-Dimensional Finite Difference Diffusion Theory Problems," ANL-82-64, Argonne National Laboratory (1984).
2. I. Dilber and E. E. Lewis, "Two Dimensional Variational Coarse Mesh Methods," *Proc. Topl. Mtg. Reactor Physics and Shielding*, Chicago, Illinois, September 17-19, 1984, p. 149, American Nuclear Society (1984).
3. I. Dilber and E. E. Lewis, "Variational Nodal Methods for Neutron Transport," *Nucl. Sci. Eng.*, **91**:132 (1985).
4. E. E. Lewis and I. Dilber, "Finite Element, Nodal and Response Matrix Methods: A Variational Synthesis for Neutron Transport," *Prog. Nucl. Energy*, **18**:63 (1986).
5. E. E. Lewis, "Interface Angular Coupling Reductions in Variational Nodal Methods for Neutron Transport," *Nucl. Sci. Eng.*, **102**:140 (1989).
6. C. B. Carrico, E. E. Lewis, and G. Palmiotti, "Three-Dimensional Variational Nodal Transport Methods for Cartesian, Triangular and Hexagonal Criticality Calculations," *Nucl. Sci. Eng.*, **111**:168 (1992).
7. G. Palmiotti, C. B. Carrico, and E. E. Lewis, "Variational Nodal Transport Methods with Anisotropic Scattering," *Nucl. Sci. Eng.*, **115**:233 (1993).
8. C. B. Carrico, U. R. Hanebutte, and E. E. Lewis, "Comparison of Space-Angle Approximations in Response Matrix Algorithms," *Proc. Jt. Int. Conf. Mathematical Methods on Supercomputing in Nuclear Applications*, **I**:58, Karlsruhe, Germany, April 19-23, 1993, Kernforschungszentrum, Karlsruhe (1993).
9. E. E. Lewis, C. B. Carrico, and G. Palmiotti, *Nucl. Sci. Eng.*, accepted for publication.
10. K. F. Laurin-Kovitz and E. E. Lewis, "Eigenvalue Perturbation Calculations for the Variational Nodal Method," *Trans. Am. Nucl. Soc.*, **70**:161 (1994).
11. C. B. Carrico, E. E. Lewis, and G. Palmiotti, "A Reduced Angular Trial Function Set for the Variational Nodal Method," *Trans. Am. Nucl. Soc.*, **65**:200 (1992).
12. R. D. Lawrence, "Progress in Nodal Methods for the Solution of the Neutron Diffusion and Transport Equations," *Prog. Nucl. Energy*, **17**:271 (1986).
13. A. Badruzzaman, "Nodal Methods in Transport Theory," *Adv. Nuclear Science and Technology*, Vol. 21, J. Lewis and M. Becker Eds., Plenum Press, New York (1990).

14. S. Wolfram, *Mathematica: A System for Doing Mathematics by Computer*, Addison-Wesley, Reading, MA (1988).
15. R. D. Lawrence, "The DIF3D Nodal Neutronics Option for Two- and Three-Dimensional Diffusion Theory Calculations in Hexagonal Geometry," ANL-83-1, Argonne National Laboratory (March 1983).
16. R. D. Lawrence, "Three-Dimensional Nodal Diffusion and Transport Methods for the Analysis of Fast-Reactor Critical Experiments," *Prog. Nucl. Energy*, **18**:1/2,101 (1986).
17. R. D. Lawrence, "Three-Dimensional Nodal Diffusion and Transport Methods for the Analysis of Fast-Reactor Critical Experiments," *Proc. Topl. Mtg. Reactor Physics and Shielding*, Chicago, Illinois, September 17-19, 1984, p. 814, American Nuclear Society (1984).
18. E. E. Lewis and W. F. Miller, Jr., *Computational Methods of Neutron Transport*, Wiley, NY, 1984.
19. V. S. Vladimirov, "Mathematical Problems in the One-Velocity of Particle Transport," Atomic Energy of Canada Ltd., Ontario, 1963: translated from V. A. Stekov *Mathematical Institute*, **61** (1961).
20. S. C. Mikhlin, *Variational Methods in Mathematical Physics*, MacMillan, NY (1964).
21. S. Lindahl and Z. Weiss, "The Response Matrix Method," *Adv. Nucl. Sci. Tech.*, **13**:72 (1981).
22. C. B. Carrico, E. E. Lewis, and G. Palmiotti, "Matrix Rank in Variational Nodal Approximations," *Trans. Am. Nucl. Soc.*, **70**:162 (1994).
23. G. Ya Rumyantsev, "Boundary Conditions in the Spherical Harmonic Method," *J. Nucl. Energy*, **16**:111 (1962).
24. P. J. Finck and K. L. Derstine, "The Application of Nodal Equivalence Theory to Hexagonal Geometry Lattices," *Int. Topl. Mtg. Adv. in Mathematics, Computations and Reactor Physics*, April 28-May 2, 1991, Pittsburgh, PA.
25. R. E. Alcouffe, *et. al.*, "User's guide for TWODANT: A Code Package for Two-Dimensional Diffusion Accelerated, Neutral Particle Transport," Los Alamos National Laboratory Report, LA-10049-M (1989).
26. W. F. Walters, *et. al.*, "TWOHEX-A Code Package for Two-Dimensional, Neutral-Particle Transport in Equilateral Triangular Meshes," *Proc. Int. Topl. Mtg. Advances in Nuclear Engineering Computational Methods*, Knoxville, Tennessee, April 9-11, 1985, **1**:164, American Nuclear Society (1985).

27. H. Khalil, T. A. Taiwo, W. S. Yang, and G. Palmiotti, "Performance of ANL Hexagonal-Geometry Nodal Diffusion Methods," *Trans. Am. Nucl. Soc.*, 71:257 (1994).
28. T. Takeda and H. Ikeda, "3-D Neutron Transport Benchmarks," NEACRP-L-330, Dept. Nucl. Engr., Osaka University, Japan (March 1991).
29. R. N. Blomquist, "VIM - A Continuous Energy Neutronics and Photon Transport Code," *Int. Topl. Mtg. Adv. in Mathematics, Computations and Reactor Physics*, April 28-May 2, 1992, Pittsburgh, PA.
30. 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.

APPENDIX A

Mathematical Scripts Used to Generate the Orthogonal Polynomials and the Submatrices needed to Calculate the Response Matrix Coefficients and the Flux Reconstruction Arrays.

ANGULAR TRIAL FUNCTIONS

(x, y, z are direction cosines)

FILE: even_parity.functions.2d

```
(*****
(*)
(*) Even Parity spherical harmonics, through P4 terms: (*)
(*) Y[0,0], Y[2,2], Y[2,1], Y[2,0], Y[4,4], Y[4,3], Y[4,2], Y[4,1], Y[4,0] (*)
(*)
(*****)
```

```
g = {
  1,
  1/2 Sqrt[15] ( y^2 - z^2 ),
  -Sqrt[15] x y,
  -1/2 Sqrt[5] ( 1 - 3 x^2 ),
  3/8 Sqrt[35] ( y^4 - 6 y^2 z^2 + z^4 ),
  -3/2 Sqrt[35/2] x y ( y^2 - 3 z^2 ),
  -3/4 Sqrt[5] ( y^2 - z^2 ) ( 1 - 7 x^2 ),
  3/2 Sqrt[5/2] x y ( 3 - 7 x^2 ),
  3/8 ( 3 - 30 x^2 + 35 x^4 ),
}
```

FILE: odd_parity.functions.2d

```
(*****
(*)
(*) Odd Parity spherical harmonics, through P5 terms with Y[n,n] deleted: (*)
(*) Y[1,0], Y[3,2], Y[3,1], Y[3,0], Y[5,4], Y[5,3], Y[5,2], Y[5,1], Y[5,0] (*)
(*)
(*****)
```

```
k = {
  Sqrt[3] x,
  1/2 Sqrt[105] x ( y^2 - z^2 ),
  1/2 Sqrt[21/2] y ( 1 - 5 x^2 ),
  1/2 Sqrt[7] x ( 5 x^2 - 3 ),
  -3/8 Sqrt[385] x ( y^4 - 6 y^2 z^2 + z^4 ),
  -1/8 Sqrt[385/2] y ( 9 x^2 - 1 ) ( y^2 - 3 z^2 ),
  1/4 Sqrt[1155] x ( 3 x^2 - 1 ) ( y^2 - z^2 ),
  -1/8 Sqrt[165] y ( 1 - 14 x^2 + 21 x^4 ),
  1/8 Sqrt[11] x ( 63 x^4 - 70 x^2 + 15 ),
}
```

FILE: even_parity.functions.3d

```
(*****
(*)
(*) Even Parity spherical harmonics, through P4 terms: (*)
(*) Y[0,0], Y[2,2], Y[2,1], Y[2,0], Y[2,-1], Y[2,-2], Y[4,4], ..., Y[4,-4] (*)
(*)
(*****)
```

```
g = {
  1,
```

```

1/2 Sqrt[15] ( y^2 - z^2 ),
-Sqrt[15] x y,
-1/2 Sqrt[5] ( 1 - 3 x^2 ),
-Sqrt[15] x z,
-Sqrt[15] y z,
3/8 Sqrt[35] ( y^4 - 6 y^2 z^2 + z^4 ),
-3/2 Sqrt[35/2] x y ( y^2 - 3 z^2 ),
-3/4 Sqrt[5] ( y^2 - z^2 ) ( 1 - 7 x^2 ),
3/2 Sqrt[5/2] x y ( 3 - 7 x^2 ),
3/8 ( 3 - 30 x^2 + 35 x^4 ),
3/2 Sqrt[5/2] x z ( 3 - 7 x^2 ),
3/2 Sqrt[5] y z ( 1 - 7 x^2 ),
3/2 Sqrt[35/2] x z ( z^2 - 3 y^2 ),
3/2 Sqrt[35] y z ( z^2 - y^2 )
}

```

FILE: odd_parity.functions.3d

```

(*****
(*)
(* Odd Parity spherical harmonics, through P5 terms with Y[n,n] deleted: *)
(* Y[1,0], Y[3,2], ..... ,Y[3,-2], Y[5,4], Y[5,3],..... , Y[5,-3], Y[5,-4] *)
(*)
(*****

```

```

k = {
  Sqrt[3] x,
  1/2 Sqrt[105] x ( y^2 - z^2 ),
  1/2 Sqrt[21/2] y ( 1 - 5 x^2 ),
  1/2 Sqrt[7] x ( 5 x^2 - 3 ),
  1/2 Sqrt[21/2] z ( 1 - 5 x^2 ),
- Sqrt[105] x y z,
-3/8 Sqrt[385] x ( y^4 - 6 y^2 z^2 + z^4 ),
-1/8 Sqrt[385/2] y ( 9 x^2 - 1 ) ( y^2 - 3 z^2 ),
  1/4 Sqrt[1155] x ( 3 x^2 - 1 ) ( y^2 - z^2 ),
-1/8 Sqrt[165] y ( 1 - 14 x^2 + 21 x^4 ),
  1/8 Sqrt[11] x ( 63 x^4 - 70 x^2 + 15 ),
-1/8 Sqrt[165] z ( 1 - 14 x^2 + 21 x^4 ),
-1/2 Sqrt[1155] x y z ( 3 x^2 - 1 ),
-1/8 Sqrt[385/2] z ( 9 x^2 - 1 ) ( 3 y^2 - z^2 ),
-3/2 Sqrt[385] x ( y^3 z - y z^3 )
}

```

(*****)

FILE: AngInt.math used in several Mathematica scripts to calculate the angular integrals. The following relations are used:

- (1) $\int \sin[x]^{(m-1)} \cos[x]^{(n-1)} dx, \{0, \pi\} = (1 + (-1)^{(m-1)}) \text{Beta}(m/2, n/2) / 2$
- (2) $\int \sin[x]^{(m-1)} \cos[x]^{(n-1)} dx, \{0, 2\pi\} = (1 + (-1)^{(m-1)} + (-1)^{(n-1)} + (-1)^{(m-1)}(-1)^{(n-1)}) \text{Beta}(m/2, n/2) / 2$

The algorithm treats each integrand as a polynomial in sines and cosines of theta and phi (where phi is the azimuthal angle). The exponents are extracted, along with any leading coefficients, and the values for the integrals in phi and theta are calculated using (1) and (2).

The integral values are summed term by term to arrive at the value of the integral of the entire integrand.

The variable "dummy" is introduced to ensure the integrand has the proper polynomial structure (i.e. the number of terms is never less than 2). "dummy" is zeroed out prior to adding the term to the sum

(*****)

```

nDig = 30
<</usr/local/math2.2/Packages/Algebra/Trigonometry.m;
AngInt [ f_ ] := Module[
  {terms, sumTerms, coeff, exp1, exp2, exp3, exp4, c1, c2, c3, c4 },
  integrand = N[ Expand[ TrigReduce[ PowerExpand[
    Sin[th]/(4 Pi) f ] ] ], nDig ] + dummy;

  (* Get the number of terms in the integrand *)
  terms      = Length[ integrand ];

  (* Calculate the integral for each term in the polynomial *)
  sumTerms = 0;
  Do[
    (*****
    Print[ "Part ",1," : ",N[ integrand[[1]] ] ];
    *****)

    (* Check to see if term is a constant *)
    If[
      NumberQ[ integrand[[1]] ],

      (* Constant term *)
      coeff = integrand[[1]];
      exp1  = 0;
      exp2  = 0;
      exp3  = 0;
      exp4  = 0,

      (* function of theta and phi *)
      If[
        Length[ integrand[[1]] ] > 1,

        If[
          NumberQ[ integrand[[1,1]] ],

          coeff = integrand[[1,1]],

          coeff = 1.0

```

```

    ],
    coeff = 1.0
];

(* Check for "dummy" structure term *)
If [ Exponent[ integrand[[1]],dummy ] > 0, coeff = 0.0 ];

(* Extract exponents of sines and cosines *)
exp1 = Exponent[ integrand[[1]],Cos[th] ];
exp2 = Exponent[ integrand[[1]],Sin[th] ];
exp3 = Exponent[ integrand[[1]],Cos[ph] ];
exp4 = Exponent[ integrand[[1]],Sin[ph] ]
];

(*****
Print[ "Cos[th] power = ",exp1 ];
Print[ "Sin[th] power = ",exp2 ];
Print[ "Cos[ph] power = ",exp3 ];
Print[ "Sin[ph] power = ",exp4 ];
Print[ "Coefficient = ",coeff ];
*****)

c1 = (1 + (-1)^exp1)/2;
c2 = (1 + (-1)^exp3 + (-1)^exp3 (-1)^exp4 + (-1)^exp4)/2;
c3 = Beta[ (exp1+1)/2, (exp2+1)/2 ];
c4 = Beta[ (exp3+1)/2, (exp4+1)/2 ];
(*****
  DEBUG
Print[ "c1, c2, c3, c4 = ",c1," ",c2," ",c3," ", c4 ];
  DEBUG
*****)
sumTerms = sumTerms + N[ coeff c1 c2 c3 c4, nDig ],
{1,terms}
];
sumTerms
]

```

```

(*****
FILE: AngIntPos.math used in the vacuum boundary condition calculation.
Same as AngInt.math except that accounts for absolute value of
direction cosine in the integrand. The following relations are used:
(1) 2*Integrate[ Sin[x]^(m-1) Cos[x]^(n-1), x, {0, Pi/2} ] =
    Beta(m/2,n/2)
(2) Integrate[ Sin[x]^(m-1) Cos[x]^(n-1), x, {0, 2 Pi} ] =
    ( 1+(-1)^(m-1)+(-1)^(n-1)+(-1)^(m-1)(-1)^(n-1) ) Beta(m/2,n/2) / 2

The algorithm treats each integrand as a polynomial in sines and cosines
of theta and phi (where phi is the azimuthal angle). The exponents
are extracted, along with any leading coefficient, and the values for
the integrals in phi and theta are calculated using (1) and (2).

The integral values are summed term by term to arrive at the value
of the integral of the entire integrand.

The variable "dummy" is introduced to ensure the integrand has the proper
polynomial structure (i.e. the number of terms is never less than 2).
"dummy" is zeroed out prior to adding the term to the sum
*****

```

```

nDig = 30
<</usr/local/math2.2/Packages/Algebra/Trigonometry.m;
AngIntPos [ f_ ] := Module[
  {terms, sumTerms, coeff, exp1, exp2, exp3, exp4, c1, c2, c3, c4 },
  integrand = N[ Expand[ TrigReduce[ PowerExpand[
    Sin[th]/(4 Pi) f ] ] ], nDig ] + dummy;

  (* Get the number of terms in the integrand *)
  terms      = Length[ integrand ];

  (* Calculate the integral for each term in the polynomial *)
  sumTerms = 0;
  Do[
    (*****
    Print[ "Part ",1," : ",N[ integrand[[1]] ] ];
    *****)

    (* Check to see if term is a constant *)
    If[
      NumberQ[ integrand[[1]] ],

      (* Constant term *)
      coeff = integrand[[1]];
      exp1  = 0;
      exp2  = 0;
      exp3  = 0;
      exp4  = 0,

      (* function of theta and phi *)
      If[
        Length[ integrand[[1]] ] > 1,

        If[
          NumberQ[ integrand[[1,1]] ],

          coeff = integrand[[1,1]],

```



```

        coeff = 1.0
    ],

    coeff = 1.0
];

(* Check for "dummy" structure term *)
If [ Exponent[ integrand[[1]],dummy ] > 0, coeff = 0.0 ];

(* Extract exponents of sines and cosines *)
exp1 = Exponent[ integrand[[1]],Cos[th] ];
exp2 = Exponent[ integrand[[1]],Sin[th] ];
exp3 = Exponent[ integrand[[1]],Cos[ph] ];
exp4 = Exponent[ integrand[[1]],Sin[ph] ]
];

(*****
Print[ "Cos[th] power = ",exp1 ];
Print[ "Sin[th] power = ",exp2 ];
Print[ "Cos[ph] power = ",exp3 ];
Print[ "Sin[ph] power = ",exp4 ];
Print[ "Coefficient   = ",coeff ];
*****)

c1    = 1;
c2    = (1 + (-1)^exp3 + (-1)^exp3 (-1)^exp4 + (-1)^exp4)/2;
c3    = Beta[ (exp1+1)/2, (exp2+1)/2 ];
c4    = Beta[ (exp3+1)/2, (exp4+1)/2 ];
(*****
  DEBUG
Print[ "c1, c2, c3, c4 = ",c1," ",c2," ",c3," ", c4 ];
  DEBUG
*****)
sumTerms = sumTerms + N[ coeff c1 c2 c3 c4, nDig ],
{1,terms}
];
sumTerms
]

```

```

(* A Mathematica script to generate the H matrix, which takes into account
the angular dependence of the node interior.
2d Geometry *)
ifa=9
rules = {x -> Cos[th], y -> Sin[th] Cos[ph], z -> Sin[th] Sin[ph]}
<<even_parity.functions.2d
g = g /. rules
o={Cos[th], Sin[th]*Cos[ph]}
<<AngInt.math
h = Table[0, {i, 2}, {j, 2}, {k, ifa}, {l, ifa}]
Do[a=AngInt[ o[[i]] o[[j]] g[[k]] g[[l]] ];
  h[[i, j, k, l]]=a;
  a=a, (*
  h[[i, j, l, k]]=a;
  h[[j, i, k, l]]=a;
  h[[j, i, l, k]]=a, *)
  {i, 2}, {j, 2}, {k, ifa}, {l, ifa}]
(* Save["hxy.dat", h] *)
stmp = OpenWrite["hxy.rawdata"]
WriteString[stmp, "H \n"]
Do[Write[stmp, N[h[[i, j, k, l]], 16]], {l, ifa}, {k, ifa}, {j, 2}, {i, 2}]
Close[stmp]

```

(*****

This MATHEMATICA script calculate the E matrix (angular coupling)
for 2-D X-Y geometry. The E matrix is defined as

$$E = \text{Integrate}[g[i] k[j] \Omega[j] \cdot n, \{th, 0, \text{Pi}\}, \{ph, 0, 2 \text{Pi}\}]$$

where $g[i]$ are the even parity angular trial functions,
 $k[j]$ are the odd parity trial functions on a given face
 n is the unit normal for the face

$\Omega \cdot n$ is always equalent to the "mu" direction in surface coordinates

The angular trial function set defined in this script covers up to a
P5 expansion.

*****)

```
ifa = 9
<<AngInt.math
<<even_parity.functions.2d
<<odd_parity.functions.2d
k = Sqrt[3] x k
k1 = k /. {x->u,y->n,z->s}
k2 = k /. {x->n,y->u,z->s}
trig1 = {x->Cos[th],y->Sin[th]Cos[ph],z->Sin[th]Sin[ph]}
trig2 = {u->Cos[th],n->Sin[th]Cos[ph],s->Sin[th]Sin[ph]}
g = g /. trig1
k1 = k1 /. trig2
k2 = k2 /. trig2
e = Table[0,{i,2},{j,ifa},{l,ifa}]
Do[
  e[[1,i,j]] = AngInt[g[[i]]k1[[j]]];
  e[[2,i,j]] = AngInt[g[[i]]k2[[j]]],
  {i,ifa},{j,ifa}
]
Save["exy.dat",e]
stmp = OpenWrite["exy.rawdata"]
WriteString[stmp,"E \n"]
Do[Write[stmp,N[e[[i,j,l]],16]],{l,ifa},{j,ifa},{i,2}]
Close[stmp]
```

(*****

This MATHEMATICA script calculate the E matrix (angular coupling)
for 2-D hexagonal geometry. The E matrix is defined as

$$E = \text{Integrate}[g[i] k[j] \Omega[j] \cdot n, \{th, 0, \text{Pi}\}, \{ph, 0, 2 \text{Pi}\}]$$

where $g[i]$ are the even parity angular trial functions,
 $k[j]$ are the odd parity trial functions on a given face
 n is the unit normal for the face

$\Omega \cdot n$ is always equalent to the "mu" direction in surface coordinates

The angular trial function set defined in this script covers up to a
P5 expansion.

*****)

(* Define number of angular moments *)

nAng = 9

(* Define even parity angular trial functions *)

<<even_parity.functions.2d

gg = g

Clear[g]

(* Define odd parity angular trial functions *)

<<odd_parity.functions.2d

kx = Sqrt[3] x k

Clear[k]

(* Rotate odd parity functions into surface coordinates *)

c1 = 1/2

c2 = Sqrt[3]/2

gg = gg /. {x -> u, y -> v, z -> w}

k1 = kx /. { x -> u, y -> v, z -> w }

k2 = kx /. { x -> c1 u + c2 v, y -> c2 u - c1 v, z -> w }

k3 = kx /. { x -> -c1 u + c2 v, y -> c2 u + c1 v, z -> w }

(* convert to direction cosines *)

trig = {u->Cos[th], v->Sin[th] Cos[ph], w->Sin[th] Sin[ph]}

gg = gg /. trig

k1 = k1 /. trig

k2 = k2 /. trig

k3 = k3 /. trig

e = Table[0, {i, 3}, {j, nAng}, {k, nAng}]

(* Calculate integrands for the E matrix *)

Print["Begin calculating v and w face data"]

<<AngInt.math

Do[

Print["Generating e[i, ", j, ", ", k, "]"];

e[[1, j, k]] = AngInt[gg[[j]] k1[[k]]];

e[[2, j, k]] = AngInt[gg[[j]] k2[[k]]];

e[[3, j, k]] = AngInt[gg[[j]] k3[[k]]],

{j, nAng}, {k, nAng}

```
]
(* Save E matrix, and generate raw data for data statement *)
(* Save["ehex.dat",e] *)
stmp = OpenWrite["ehex.rawdata"]
WriteString[stmp,"E \n"]
Do[
  Write[ stmp, Chop[ N[ e[[i,j,k]], 12 ], 10^-14 ] ],
  {k,nAng},{j,nAng},{i,3}
]
Close[stmp]
```

```

(* A Mathematica script to generate the V matrix,
   the angular dependence array needed in the anisotropic scattering
   calculation. 2D Geometry *)
(* Define even parity angular trial functions *)
<<even_parity.functions.2d
gg = g
Clear[g]
(* Define odd-parity scattering functions *)
gm = {Sqrt[3]x,
      Sqrt[3]y,
      Sqrt[7/4] (5x^3-3x),
      Sqrt[21/8] (5x^2-1)y,
      Sqrt[105/4] (y^2-z^2)x,
      Sqrt[35/8] (y^2-3z^2)y,
      1/8 Sqrt[11] x ( 63 x^4 - 70 x^2 + 15 ),
      -1/8 Sqrt[165] y ( 1 - 14 x^2 + 21 x^4 ),
      1/4 Sqrt[1155] x ( 3 x^2 - 1 ) ( y^2 - z^2 ),
      -1/8 Sqrt[385/2] y ( 9 x^2 - 1 ) ( y^2 - 3 z^2 ),
      -3/8 Sqrt[385] x ( y^4 - 6 y^2 z^2 + z^4 ),
      3/8 Sqrt[77/2] y ( y^4 - 10 y^2 z^2 + 5 z^4 ),
      }
o={Cos[th], Sin[th]*Cos[ph]}

trig = {x->Cos[th], y->Sin[th]Cos[ph], z->Sin[th]Sin[ph]}
gg = gg /. trig
gm = gm /. trig

v = Table[0, {i, 2}, {j, 9}, {k, 12}]

(* Calculate integrands for the V matrix *)
<<AngInt.math
Do[
  Print[ "Generating v[i, ", j, ", ", k, "]" ];
  v[[1, j, k]] = AngInt[ o[[1]]gg[[j]] gm[[k]] ];
  v[[2, j, k]] = AngInt[ o[[2]]gg[[j]] gm[[k]] ],
  {j, 9}, {k, 12}
]

(* Save V matrix, and generate raw data for data
statement *)
(* Save["vxy.dat",v] *)
stmp = OpenWrite["vxy.rawdata"]
WriteString[stmp, "V \n"]
Do[
  Write[ stmp, Chop[ N[ v[[i, j, k]], 12 ], 10^-14 ] ],
  {k, 12}, {j, 9}, {i, 2}
]
Close[stmp]

```

```

(* A Mathematica script to generate vacuum boundary conditions.
  2D Geometry *)

r1 = {x -> Cos[th], y -> Sin[th] Cos[ph], z -> Sin[th] Sin[ph]}
r2 = {u -> Cos[th], n -> Sin[th] Cos[ph], s -> Sin[th] Sin[ph]}

<<even_parity.functions.2d
g1 = g /. r1

<<AngIntPos.math

<<exy.dat;
e1 = Table[ e[[1,i,j]], {i,9}, {j,9}];
l  = Table[0, {j,9}, {k,9}];
uu  = Table[0, {j,9}, {k,9}];
vac = Table[0, {j,9}, {k,9}];
i9 = IdentityMatrix[9];

Do[
  l[[j,k]] = AngIntPos[ Cos[th] g1[[j]] g1[[k]] l,
    {j,9}, {k,9}
  ];
uu = (Transpose[e1].Inverse[l].e1)/2;
vac = Inverse[uu+i9].(uu-i9);
(* Save["vacxy.dat", e1, l, uu, vac]

stmp = OpenWrite["vacxy.out", FormatType->OutputForm]
Write[stmp, MatrixForm[N[vac,8]]]
Close[stmp] *)

stmp = OpenWrite["vacxy.rawdata"]
WriteString[stmp, "P \n"]
Do[Write[stmp, N[vac[[i,j]], 16]], {j,9}, {i,9}]
Close[stmp]

```

```

(* A Mathematica script to generate the H matrix, which takes into account
the angular dependence of the node interior.
3d Geometry *)
rules = {x -> Cos[th], y -> Sin[th]Cos[ph], z -> Sin[th]Sin[ph]}
<<even_parity.functions.3d
<<AngInt.math
g = g /. rules
o={Cos[th],Sin[th]*Cos[ph],Sin[th] Sin[ph]}
h = Table[0, {i,3}, {j,3}, {k,15}, {l,15}]
Do[
  a=AngInt[ o[[i]] o[[j]] g[[k]] g[[l]] ];
  h[[i,j,k,l]]=a;
  h[[i,j,l,k]]=a;
  h[[j,i,k,l]]=a;
  h[[j,i,l,k]]=a,
  {i,3}, {j,i,3}, {k,15}, {l,k,15}]
(* Save["hxyz.dat",h] *)
stmp = OpenWrite["hxyz.rawdata"]
WriteString[stmp,"H \n"]
Do[Write[stmp,N[h[[i,j,k,l]],16]],{l,15},{k,15},{j,3},{i,3}]
Close[stmp]

```


(*****

This MATHEMATICA script calculate the E matrix (angular coupling)
for 3-D X-Y-Z geometry. The E matrix is defined as

$$E = \text{Integrate}[g[i] k[j] \Omega[j] \cdot n, \{th, 0, \text{Pi}\}, \{ph, 0, 2 \text{Pi}\}]$$

where $g[i]$ are the even parity angular trial functions,
 $k[j]$ are the odd parity trial functions on a given face
 n is the unit normal for the face

$\Omega \cdot n$ is always equalent to the "mu" direction in surface coordinates

The angular trial function set defined in this script covers up to a
P5 expansion.

*****)

```
ifa = 15
<<AngInt.math
<<even_parity.functions.3d
<<odd_parity.functions.3d
k = Sqrt[3] x k
k1 = k /. {x->u,y->n,z->s}
k2 = k /. {x->n,y->u,z->s}
k3 = k /. {x->s,y->u,z->n}
trig1 = {x->Cos[th],y->Sin[th]Cos[ph],z->Sin[th]Sin[ph]}
trig2 = {u->Cos[th],n->Sin[th]Cos[ph],s->Sin[th]Sin[ph]}
g = g /. trig1
k1 = k1 /. trig2
k2 = k2 /. trig2
k3 = k3 /. trig2
e = Table[0,{i,3},{j,ifa},{l,ifa}]
Do[
  e[[1,i,j]] = AngInt[g[[i]]k1[[j]]];
  e[[2,i,j]] = AngInt[g[[i]]k2[[j]]];
  e[[3,i,j]] = AngInt[g[[i]]k3[[j]]],
  {i,ifa},{j,ifa}
]
(* Save["exyz.dat",e] *)
stmp = OpenWrite["exyz.rawdata"]
WriteString[stmp,"E \n"]
Do[Write[stmp,N[e[[i,j,1]],16]],{l,ifa},{j,ifa},{i,3}]
Close[stmp]
```

(*****)

This MATHEMATICA script calculate the E matrix (angular coupling)
for 3-D hexagonal geometry. The E matrix is defined as

$$E = \text{Integrate}[g[i] k[j] \Omega[j] \cdot n, \{th, 0, \text{Pi}\}, \{ph, 0, 2 \text{ Pi}\}]$$

where $g[i]$ are the even parity angular trial functions,
 $k[j]$ are the odd parity trial functions on a given face
 n is the unit normal for the face

$\Omega \cdot n$ is always equalent to the "mu" direction in surface coordinates

The angular trial function set defined in this script covers up to a
P5 expansion.

(*****)

nAng = 15

<<AngInt.math

<<even_parity.functions.3d

<<odd_parity.functions.3d

gg = g

Clear[g]

kx = Sqrt[3] x k

Clear[k]

(* Rotate odd parity functions into surface coordinates *)

c1 = 1/2

c2 = Sqrt[3]/2

gg = gg /. {x -> u, y -> v, z -> w}

k1 = kx /. { x -> u, y -> v, z -> w }

k2 = kx /. {x -> c1 u + c2 v, y -> c2 u - c1 v, z -> w }

k3 = kx /. { x -> -c1 u + c2 v, y -> c2 u + c1 v, z -> w }

k4 = kx /. { x -> w, y -> u, z -> v }

(* convert to direction cosines *)

trig = {u->Cos[th], v->Sin[th] Cos[ph], w->Sin[th] Sin[ph]}

gg = gg /. trig

k1 = k1 /. trig

k2 = k2 /. trig

k3 = k3 /. trig

k4 = k4 /. trig

Clear[e]

e = Table[0, {i, 4}, {j, nAng}, {k, nAng}]

Do[

Print["Generating e[i, ", j, ", ", k, "]"];

e[[1, j, k]] = AngInt[gg[[j]] k1[[k]]];

e[[2, j, k]] = AngInt[gg[[j]] k2[[k]]];

e[[3, j, k]] = AngInt[gg[[j]] k3[[k]]];

e[[4, j, k]] = AngInt[gg[[j]] k4[[k]]],

{j, nAng}, {k, nAng}

]

```
(* Save E matrix, and generate raw data for data statement *)
(* Save["ehexz.dat",e] *)
stmp = OpenWrite["ehexz.rawdata"]
WriteString[stmp,"E \n"]
Do[
  Write[ stmp, Chop[ N[ e[[i,j,k]], 12 ], 10^-14 ] ],
  {k,nAng},{j,nAng},{i,4}
]
Close[stmp]
```

```

(* A Mathematica script to generate the V matrix,
   the angular dependence array needed in the anisotropic scattering
   calculation. 3D Geometry *)
trig = {x -> Cos[th], y -> Sin[th] Cos[ph], z -> Sin[th] Sin[ph]}
(* Define even parity angular trial functions *)
<<even_parity.functions.3d
gp = g
Clear[g]
gp = gp /. trig

(* Define odd-parity scattering functions *)
gm = {Sqrt[3]x,
      Sqrt[3]y,
      Sqrt[3]z,
      Sqrt[7/4] (5x^3-3x),
      Sqrt[21/8] (5x^2-1)y,
      Sqrt[21/8] (5x^2-1)z,
      Sqrt[105/4] (y^2-z^2)x,
      Sqrt[105]x*y*z,
      Sqrt[35/8] (y^2-3z^2)y,
      Sqrt[35/8] (3y^2-z^2)z,
      1/8 Sqrt[11] x ( 63 x^4 - 70 x^2 + 15 ),
      -1/8 Sqrt[165] y ( 1 - 14 x^2 + 21 x^4 ),
      -1/8 Sqrt[165] z ( 1 - 14 x^2 + 21 x^4 ),
      1/4 Sqrt[1155] x ( 3 x^2 - 1 ) ( y^2 - z^2 ),
      -1/2 Sqrt[1155] x y z ( 3 x^2 - 1 ),
      -1/8 Sqrt[385/2] y ( 9 x^2 - 1 ) ( y^2 - 3 z^2 ),
      -1/8 Sqrt[385/2] z ( 9 x^2 - 1 ) ( 3 y^2 - z^2 ),
      -3/8 Sqrt[385] x ( y^4 - 6 y^2 z^2 + z^4 ),
      -3/2 Sqrt[385] x ( y^3 z - y z^3 ),
      3/8 Sqrt[77/2] y ( y^4 - 10 y^2 z^2 + 5 z^4 ),
      3/8 Sqrt[77/2] z ( z^4 - 10 y^2 z^2 + 5 y^4 ),
      }

gm = gm /. trig

o={Cos[th], Sin[th]*Cos[ph], Sin[th] Sin[ph]}

<< AngInt.math

v = Table[0, {j, 3}, {k, 15}, {l, 21}]
Do[v[[j,k,l]]=AngInt[o[[j]]gp[[k]]gm[[l]]], {j, 3}, {k, 15}, {l, 21}]
(* Save["vxyz.dat", v] *)
Put[V, "vxyz.rawdata"]
Do[PutAppend[N[v[[j,k,l]], 16], "vxyz.rawdata"], {l, 21}, {k, 15}, {j, 3}]

```

```

(* A Mathematica script to generate vacuum boundary conditions.
   3D Geometry *)
ifa = 15
r1 = {x -> Cos[th], y -> Sin[th] Cos[ph], z -> Sin[th] Sin[ph]}
r2 = {u -> Cos[th], n -> Sin[th] Cos[ph], s -> Sin[th] Sin[ph]}
<< even_parity.functions.3d
g1 = g /. r1
<<AngIntPos.math

<<exyz.dat;
e1 = Table[ e[[1,i,j]], {i,ifa}, {j,ifa}];
l   = Table[0, {j,ifa}, {k,ifa}];
uu  = Table[0, {j,ifa}, {k,ifa}];
vacz = Table[0, {j,ifa}, {k,ifa}];
i4 = IdentityMatrix[ifa];

Do[
  l[[j,k]] = AngIntPos[ Cos[th] g1[[j]] g1[[k]] ],
  {j,ifa}, {k,ifa}
];
Print["Inverting matrix"];
uu = (Transpose[e1].Inverse[l].e1)/2 ;
vacz = Inverse[uu+i4].(uu-i4);
(* Save["vacxyz.dat", e1, l, uu, vacz]

stmp = OpenWrite["vacxyz.out", FormatType->OutputForm]
Write[stmp, MatrixForm[N[vacz, 8]]]
Close[stmp] *)

stmp = OpenWrite["vacxyz.rawdata"]
WriteString[stmp, "P \n"]
Do[Write[stmp, N[vacz[[i,j]], 16]], {j,ifa}, {i,ifa}]
Close[stmp]

```

SPATIAL TRIAL FUNCTIONS

FILE: f.surf.xy.dat

Trial functions on a side of a X-Y node. Expansion order: 2

$$l = \{1, 2*3^{(1/2)}*x, -5^{(1/2)}/2 + 6*5^{(1/2)}*x^2\}$$

FILE: f.vol.xy.dat

Trial functions on the interior of a X-Y node. Expansion order: 4

$$f = \{1, 2*3^{(1/2)}*x, 2*3^{(1/2)}*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*x^2, 12*x*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*y^2, -3*7^{(1/2)}*x + 20*7^{(1/2)}*x^3, -(15^{(1/2)}*y) + 12*15^{(1/2)}*x^2*y, -(15^{(1/2)}*x) + 12*15^{(1/2)}*x*y^2, -3*7^{(1/2)}*y + 20*7^{(1/2)}*y^3, -21/8 + 210*x^4 - (3*5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*x^2))/2, -6*21^{(1/2)}*x*y + 40*21^{(1/2)}*x^3*y, -5/4 - (5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*x^2))/2 + 180*x^2*y^2 - (5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*y^2))/2, -6*21^{(1/2)}*x*y + 40*21^{(1/2)}*x*y^3, -21/8 + 210*y^4 - (3*5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*y^2))/2\}$$

FILE: f.surf.hex.dat

Trial functions on a side of a hex node. Expansion order: 2

$$l = \{1, 3*2^{(1/2)}*3^{(1/4)}*x, -5^{(1/2)}/2 + 9*15^{(1/2)}*x^2\}$$

FILE: f.vol.hex.dat

Trial functions on the interior of an hex node. Expansion order: 6

$$f = \{1., 3.531397147659254*x, 3.531397147659254*y, -0.992094737665681 + 12.37218113922247*x^2, 15.2127765851133*x*y, -1.386623516201175 + 4.220134183810281*x^2 + 13.07212295960745*y^2, -7.364172208855169*x + 45.55400150508527*x^3, -2.843918214276882*y + 52.77654471105122*x^2*y, -6.215410878336123*x + 15.34253644105606*x^3 + 69.31604172534397*x*y^2, -9.20730654962232*y + 30.49051331999612*x^2*y + 46.79193393431396*y^3, 1.144405667459047 - 40.06601247254548*x^2 + 168.2602416904509*x^4 - 1.406229436608328*y^2, -26.50296143625209*x*y + 193.4898333051816*x^3*y, 0.937907092856306 - 25.07947109327715*x^2 + 54.30072427626424*x^4 - 8.90944642896137*y^2 + 250.7950289971609*x^2*y^2, -50.42654572628691*x*y + 130.7053493868987*x^3*y + 302.4959930449739*x*y^3, 1.539474445844345 - 8.59954682742715*x^2 + 5.127712221713981*x^4 - 47.18963566829999*y^2 + 173.7862848738076*x^2*y^2 + 163.2901353147327*y^4, 11.6653537350933*x - 194.1305451511002*x^3 + 623.0154297905012*x^5 - 5.717241004090825*x*y^2, 4.69185605425132*y - 157.7548620476962*x^2*y + 717.9223962484047*x^4*y - 9.2171369165954*y^3, 9.69755937019898*x - 106.9840393103204*x^3 + 197.8190140565777*x^5 - 102.6298897072043*x*y^2 + 917.106990708609*x^3*y^2, 5.264241230760947*y - 205.4054425941534*x^2*y + 464.0234379739513*x^4*y - 27.72465200112097*y^3 + 1184.265472234982*x^2*y^3,\}$$

309.2541675819772*x*y^2 + 851.414464715421*x^3*y^2 +
 1255.463567163217*x*y^4, 15.68457470472548*y - 111.28234482202*x^2*y +
 132.6642324402051*x^4*y - 215.3509727206996*y^3 +
 899.228572510356*x^2*y^3 + 561.3590955770824*y^5,
 -1.060095787822545 + 83.019683621295*x^2 - 889.228572832441*x^4 +
 2313.052962198468*x^6 - 1.293105354507158*y^2 -
 11.53703521761299*x^2*y^2 + 9.39119022184615*y^4,
 0.0000108358095451932*x*y*(4.335238300855329*10^6 - 7.246427*10^7*x^2 +
 2.452554741140914*10^8*x^4 - 4.901970000000001*10^5*y^2),
 -1.342920857770678 + 63.30661830660211*x^2 - 448.8548508855424*x^4 +
 720.0866810090347*x^6 + 21.19039857921666*y^2 -
 682.8485950113241*x^2*y^2 + 3419.545944882732*x^4*y^2 -
 41.92742357206419*y^4, 66.87661377195044*x*y - 878.806782568358*x^3*y +
 1698.779184200614*x^5*y - 388.6964335735138*x*y^3 +
 4389.10418853506*x^3*y^3, -0.69529255111581 + 33.14469660668448*x^2 -
 108.7042593792074*x^4 + 17.72340254766733*x^6 + 23.12292942763903*y^2 -
 1326.755058402553*x^2*y^2 + 3138.811085482705*x^4*y^2 -
 84.7848146373795*y^4 + 5426.412021403232*x^2*y^4,
 100.3195658795806*x*y - 505.5173873502208*x^3*y +
 447.2929180746544*x^5*y - 1647.523967451926*x*y^3 +
 4884.832425884538*x^3*y^3 + 4986.581349244301*x*y^5,
 -1.735740046497104 + 18.46246594573035*x^2 - 45.48807275055156*x^4 +
 28.55381163561693*x^6 + 106.555405180707*y^2 - 891.974915726297*x^2*y^2 +
 1307.292857878233*x^4*y^2 - 918.732525582574*y^4 +
 4380.113505445939*x^2*y^4 + 1915.895220021416*y^6]

FILE: f.surfx.yz.dat

Trial functions on a surface of a X-Y-Z node. Expansion order: 2

$$f = \{1, 2*3^{(1/2)}*x, 2*3^{(1/2)}*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*x^2, 12*x*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*y^2\}$$

FILE: f.vol.yz.dat

Trial functions on the interior of a X-Y-Z node. Expansion order: 4

$$f = \{1, 2*3^{(1/2)}*x, 2*3^{(1/2)}*y, 2*3^{(1/2)}*z, -5^{(1/2)}/2 + 6*5^{(1/2)}*x^2, 12*x*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*y^2, 12*y*z, -5^{(1/2)}/2 + 6*5^{(1/2)}*z^2, 12*x*z, -3*7^{(1/2)}*x + 20*7^{(1/2)}*x^3, -(15^{(1/2)}*y) + 12*15^{(1/2)}*x^2*y, -(15^{(1/2)}*x) + 12*15^{(1/2)}*x*y^2, -3*7^{(1/2)}*y + 20*7^{(1/2)}*y^3, -(15^{(1/2)}*z) + 12*15^{(1/2)}*y^2*z, -(15^{(1/2)}*y) + 12*15^{(1/2)}*y*z^2, -3*7^{(1/2)}*z + 20*7^{(1/2)}*z^3, -(15^{(1/2)}*x) + 12*15^{(1/2)}*x*z^2, -(15^{(1/2)}*z) + 12*15^{(1/2)}*x^2*z, 24*3^{(1/2)}*x*y*z, -1/(80*(1/3600 - (-1/(32*5^{(1/2)})) + (3*5^{(1/2)})/224)^2)^{(1/2)} + x^4/(1/3600 - (-1/(32*5^{(1/2)})) + (3*5^{(1/2)})/224)^2)^{(1/2)} - ((-1/(32*5^{(1/2)})) + (3*5^{(1/2)})/224)*(-5^{(1/2)}/2 + 6*5^{(1/2)}*x^2)/(1/3600 - (-1/(32*5^{(1/2)})) + (3*5^{(1/2)})/224)^2)^{(1/2)}, -6*21^{(1/2)}*x*y + 40*21^{(1/2)}*x^3*y, -1/(144*(1/14400 - (1/(32*5^{(1/2)})) - 5^{(1/2)}/288)^2)^{(1/2)} - ((1/(32*5^{(1/2)})) - 5^{(1/2)}/288)*(-5^{(1/2)}/2 + 6*5^{(1/2)}*x^2)/(1/14400 - (1/(32*5^{(1/2)})) - 5^{(1/2)}/288)^2)^{(1/2)} + (x^2*y^2)/(1/14400 - (1/(32*5^{(1/2)})) - 5^{(1/2)}/288)^2)^{(1/2)} - (-5^{(1/2)}/2 + 6*5^{(1/2)}*y^2)/(72*5^{(1/2)}*(1/14400 - (1/(32*5^{(1/2)})) - 5^{(1/2)}/288)^2)^{(1/2)}, -6*21^{(1/2)}*x*y + 40*21^{(1/2)}*x*y^3, -21/8 + 210*y^4 - (3*5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*y^2))/2,$$

$$\begin{aligned}
& -6*21^{(1/2)}*y*z + 40*21^{(1/2)}*y^3*z, \\
& -5/4 - (5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*y^2))/2 + 180*y^2*z^2 - \\
& (5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*z^2))/2, \\
& -6*21^{(1/2)}*y*z + 40*21^{(1/2)}*y*z^3, \\
& -21/8 + 210*z^4 - (3*5^{(1/2)}*(-5^{(1/2)}/2 + 6*5^{(1/2)}*z^2))/2, \\
& -6*21^{(1/2)}*x*z + 40*21^{(1/2)}*x*z^3, \\
& -1/(144*(1/14400 - (1/(32*5^{(1/2)}) - 5^{(1/2)}/288)^2)^{(1/2)}) - \\
& ((1/(32*5^{(1/2)}) - 5^{(1/2)}/288)*(-5^{(1/2)}/2 + 6*5^{(1/2)}*x^2))/ \\
& (1/14400 - (1/(32*5^{(1/2)}) - 5^{(1/2)}/288)^2)^{(1/2)} + \\
& (x^2*z^2)/(1/14400 - (1/(32*5^{(1/2)}) - 5^{(1/2)}/288)^2)^{(1/2)} - \\
& (-5^{(1/2)}/2 + 6*5^{(1/2)}*z^2)/ \\
& (72*5^{(1/2)}*(1/14400 - (1/(32*5^{(1/2)}) - 5^{(1/2)}/288)^2)^{(1/2)}), \\
& -6*21^{(1/2)}*x*z + 40*21^{(1/2)}*x^3*z, -6*5^{(1/2)}*y*z + 72*5^{(1/2)}*x^2*y*z, \\
& -6*5^{(1/2)}*x*z + 72*5^{(1/2)}*x*y^2*z, -6*5^{(1/2)}*x*y + 72*5^{(1/2)}*x*y*z^2
\end{aligned}$$

FILE: f.surfx.hexz.dat

Trial functions on a X-Y surface of a hex-Z node. Expansion order: 2

$$f = \{1, 3*2^{(1/2)}*3^{(1/4)}*x, 2*3^{(1/2)}*y, -5^{(1/2)}/2 + 9*15^{(1/2)}*x^2, 6*2^{(1/2)}*3^{(3/4)}*x*y, -5^{(1/2)}/2 + 6*5^{(1/2)}*y^2\}$$

FILE: f.surfh.hexz.dat

Trial functions on a hex surface of a hex-Z node. Expansion order: 2

$$\begin{aligned}
f = \{ & 1, (6*3^{(1/4)}*x)/5^{(1/2)}, (6*3^{(1/4)}*y)/5^{(1/2)}, \\
& -5*(5/127)^{(1/2)} + 36*(15/127)^{(1/2)}*x^2, 18*(5/7)^{(1/2)}*x*y, \\
& (-5*(635/903)^{(1/2)})/4 + (41*(-5*(5/127)^{(1/2)} + 36*(15/127)^{(1/2)}*x^2))/ \\
& (4*903^{(1/2)}) + 9*(635/301)^{(1/2)}*y^2\}
\end{aligned}$$

FILE: f64.vol.hexz.dat

Trial functions on the interior of a hex-Z node.
Expansion order: 6 in the X-Y plane, 4 in the Z axis.

$$\begin{aligned}
f = \{ & 1., 3.531397147659254*x, 3.531397147659254*y, 3.464101615137754*z, \\
& -0.992094737665681 + 12.37218113922247*x^2, 15.2127765851133*x*y, \\
& -1.386623516201175 + 4.22013418381028*x^2 + 13.07212295960745*y^2, \\
& 12.23311856289929*y*z, -1.118033988749895 + 13.41640786499874*z^2, \\
& 12.23311856289929*x*z, -7.364172208855168*x + 45.55400150508527*x^3, \\
& -2.843918214276883*y + 52.77654471105124*x^2*y, \\
& -6.215410878336125*x + 15.34253644105606*x^3 + 69.31604172534398*x*y^2, \\
& -9.20730654962232*y + 30.49051331999613*x^2*y + 46.79193393431396*y^3, \\
& -3.436716983117353*z + 42.85849266715744*y^2*z, \\
& -3.948222038857477*y + 47.37866446628973*y*z^2, \\
& -7.937253933193773*z + 52.91502622129182*z^3, \\
& -3.948222038857477*x + 47.37866446628973*x*z^2, \\
& -4.803404762060481*z + 45.28316225765551*x^2*z + 14.61897364223524*y^2*z, \\
& 52.6986039392208*x*y*z, 1.144405667459047 - 40.06601247254547*x^2 + \\
& 168.2602416904509*x^4 - 1.406229436608328*y^2, \\
& -26.5029614362521*x*y + 193.4898333051816*x^3*y, \\
& 0.937907092856306 - 25.07947109327715*x^2 + 54.30072427626425*x^4 - \\
& 8.90944642896137*y^2 + 250.7950289971609*x^2*y^2, \\
& -50.42654572628692*x*y + 130.7053493868988*x^3*y + \\
& 302.4959930449739*x*y^3, 1.539474445844345 - 8.59954682742714*x^2 + \\
& 5.127712221713985*x^4 - 47.18963566829997*y^2 + \\
& 173.7862848738076*x^2*y^2 + 163.2901353147327*y^4,
\end{aligned}$$

-22.68916516138272*y*z + 140.3528101458449*y^3*z,
 1.109195636770142 + 8.88178419700125*10^-16*x^2 - 13.83251902862112*y^2 -
 13.31034764124171*z^2 + 165.9902283434534*y^2*z^2,
 -28.02959589992768*y*z + 186.8639726661845*y*z^3,
 1.125 - 45.0000000000001*z^2 + 210.*z^4,
 -28.02959589992768*x*z + 186.8639726661845*x*z^3,
 1.550292220712804 - 14.615077773959*x^2 - 4.718253454585189*y^2 -
 18.60350664855365*z^2 + 175.380933287508*x^2*z^2 +
 56.61904145502227*y^2*z^2, -25.51024084284776*x*z +
 157.8036901897536*x^3*z, -24.48571340145234*y*z +
 211.1407608441662*x^2*y*z + 81.0858166384408*y^3*z,
 -21.53081486238894*x*z + 53.14810526577217*x^3*z +
 240.1178120957201*x*y^2*z, -17.00840128541522*x*y +
 204.1008154249827*x*y*z^2, 11.66535373509326*x - 194.1305451510995*x^3 +
 623.0154297904985*x^5 - 5.717241004090801*x*y^2,
 4.691856054251322*y - 157.7548620476962*x^2*y + 717.9223962484051*x^4*y -
 9.2171369165954*y^3, 9.69755937019897*x - 106.9840393103202*x^3 +
 197.8190140565769*x^5 - 102.6298897072043*x*y^2 +
 917.106990708609*x^3*y^2, 5.264241230760949*y -
 205.4054425941535*x^2*y + 464.0234379739516*x^4*y -
 27.72465200112097*y^3 + 1184.265472234982*x^2*y^3,
 8.09990722997389*x - 32.6914364353943*x^3 + 9.0669661188542*x^5 -
 309.2541675819773*x*y^2 + 851.414464715421*x^3*y^2 +
 1255.463567163217*x*y^4, 15.68457470472548*y - 111.28234482202*x^2*y +
 132.8642324402053*x^4*y - 215.3509727206996*y^3 +
 899.228572510356*x^2*y^3 + 561.3590955770824*y^5,
 -1.060095787822544 + 83.019683621295*x^2 - 889.228572832441*x^4 +
 2313.052962198468*x^6 - 1.293105354507159*y^2 -
 11.53703521761299*x^2*y^2 + 9.39119022184614*y^4,
 46.97581656109533*x*y - 785.2090285514575*x^3*y +
 2657.541607416357*x^5*y - 53.11681331625073*x*y^3,
 -1.342920857770676 + 63.30661830660207*x^2 - 448.8548508855424*x^4 +
 720.0866810090344*x^6 + 21.19039857921664*y^2 -
 682.8485950113239*x^2*y^2 + 3419.545944882731*x^4*y^2 -
 41.92742357206416*y^4, 66.87661377195044*x*y - 878.806782568358*x^3*y +
 1698.779184200614*x^5*y - 388.6964335735138*x*y^3 +
 4389.10418853506*x^3*y^3, -0.6952925511158102 + 33.14469660668446*x^2 -
 108.7042593792074*x^4 + 17.72340254766721*x^6 + 23.12292942763903*y^2 -
 1326.755058402552*x^2*y^2 + 3138.811085482704*x^4*y^2 -
 84.7848146373794*y^4 + 5426.412021403232*x^2*y^4,
 100.3195658795807*x*y - 505.5173873502208*x^3*y +
 447.2929180746549*x^5*y - 1647.523967451927*x*y^3 +
 4884.832425884538*x^3*y^3 + 4986.581349244301*x*y^5,
 -1.735740046497101 + 18.46246594573032*x^2 - 45.4880727505516*x^4 +
 28.55381163561682*x^6 + 106.5554051807069*y^2 -
 891.974915726297*x^2*y^2 + 1307.292857878232*x^4*y^2 -
 918.732525582574*y^4 + 4380.113505445938*x^2*y^4 + 1915.895220021416*y^6}

```
(* A Mathematica script to generate orthonormal trial functions
over an XY node
```

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as 1/sqrt(volume) (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \text{sum}(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\text{sqrt}(\langle g[n],g[n] \rangle - \text{sum}(\langle f[i],g[n] \rangle^2:i=1,n-1)) \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n] \rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

```
*****)
```

```
(* Define the volume integral over the node *)
```

```
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}]
```

```
(* Define a vector consisting of the functions making up a complete
fourth order polynomial *)
```

```
g = {1,x,y,x^2,x*y,y^2,x^3,x^2*y,x*y^2,y^3,x^4,x^3*y,x^2*y^2,x*y^3,y^4}
```

```
(* Define and initialize a vector for the orthogonal trial functions *)
```

```
f = Table[0, {i, 15}]
```

```
(* Define and initialize a vector for the trial function coefficients *)
```

```
a = Table[0, {i, 15}]
```

```
(* Define the first trial function as 1 *)
```

```
f[[1]] = 1
```

```
(* Begin loop to determine trial functions *)
```

```
Do[
```

```
  Print["Generating trial function ", n];
```

```
(* Calculate inner products *)
```

```
  Do[ a[[j]] = VolInt[f[[j]]*g[[n]], {j, 1, n-1} ];
```

```
  a[[n]] = VolInt[g[[n]]*g[[n]]];
```

```
(* Calculate sum of squares of inner products *)
```

```

sum = 0;
Do[ sum = sum+a[[j]]^2,{j,n-1} ];

(* Calculate the values of the coefficients *)

a[[n]] = 1/Sqrt[a[[n]]-sum];
Do[ a[[j]] = -a[[n]]*a[[j]], {j,n-1} ];

(* Store the trial function in f[n] *)

sum = 0;
Do[ sum = sum+a[[j]]*f[[j]],{j,n-1} ];
f[[n]] = sum + a[[n]]*g[[n]],

(* End of Do loop *)
{n,2,15} ]

(* Save the set of trial functions *)

Save["f.vol.xy.dat",f]

```

```

(* A Mathematica script to generate orthonormal trial functions
   over an XY side *)
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}]
g = {1, x, x^2}
l = Table[0, {i, 3}]
a = Table[0, {i, 3}]
l[[1]] = 1
Do[
  Print["Generating trial function ", n];
  Do[ a[[j]] = VolInt[l[[j]]*g[[n]], {j, 1, n-1} ];
  a[[n]] = VolInt[g[[n]]*g[[n]];
  Print["End of VolInt"];
  sum = 0;
  Do[ sum = sum+a[[j]]^2, {j, n-1} ];
  a[[n]] = 1/Sqrt[a[[n]]-sum];
  Do[ a[[j]] = -a[[n]]*a[[j]], {j, n-1} ];
  sum = 0;
  Do[ sum = sum+a[[j]]*l[[j]], {j, n-1} ];
  l[[n]] = sum + a[[n]]*g[[n]],
  {n, 2, 3} ]
Save["f.surf.xy.dat", l]

```

```

(* A Mathematica script to generate the P matrix, which takes into account
the spatial dependence of the node interior.
X-Y Geometry *)
<<f.vol.xy.dat
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}]
s = {x, y}
p = Table[0, {i, 15}, {j, 15}, {k, 2}, {l, 2}]
Do[
  If[k==1,
    Do[p[[{i, j, k, l}] = VolInt[D[f[[{i}], s[[k]]]*D[f[[{j}], s[[l]]]]];
      Print[i, " ", j, " ", k, " ", l, " ", p[[{i, j, k, l}]]];
      p[[{j, i, k, l}] = p[[{i, j, k, l}],
        {i, 15}, {j, i, 15}
    ],
    Do[p[[{i, j, k, l}] = VolInt[D[f[[{i}], s[[k]]]*D[f[[{j}], s[[l]]]]];
      Print[i, " ", j, " ", k, " ", l, " ", p[[{i, j, k, l}]]];
      p[[{j, i, l, k}] = p[[{i, j, k, l}],
        {i, 15}, {j, 15}
    ]
  ],
  {k, 2}, {l, k, 2}
]
(* Save["pxy.dat", p] *)
Put["P", "pxy.rawdata"]
Do[PutAppend[N[p[[{i, j, k, l}], 16], "pxy.rawdata"], {1, 2}, {k, 2}, {j, 15}, {i, 15}]

```

(* A Mathematica script to generate the D matrix, which couples
the spatial dependence of the node surfaces to the node interior.
X-Y Geometry

2

```
*****
*
4 *           * 1
*           *
*****
```

3

```
*)
<<f.vol.xy.dat
<<f.surf.xy.dat
l = 1 /. x->s
f1 = f /. {x->1/2,y->s}
f2 = f /. {y->1/2,x->s}
mx = {1,-1,1,1,-1,1,-1,1,-1,1,1,-1,1,-1,1}
my = {1,1,-1,1,-1,1,1,-1,1,-1,1,-1,1,-1,1}
SurfInt[f_] := Integrate[f,{s,-1/2,1/2}]
d = Table[0,{i,4},{j,15},{k,3}]
Do[ d[[1,j,k]] = SurfInt[f1[[j]]*l[[k]]];
    Print[" side 1 ",d[[1,j,k]]];
    d[[3,j,k]] = mx[[j]]*d[[1,j,k]];
    Print[" side 3 ",d[[3,j,k]]];
    d[[2,j,k]] = SurfInt[f2[[j]]*l[[k]]];
    Print[" side 2 ",d[[2,j,k]]];
    d[[4,j,k]] = my[[j]]*d[[2,j,k]];
    Print[" side 4 ",d[[4,j,k]]],
    {j,15},{k,3}
]
(* Save["dxy.dat",d] *)
Put[D,"dxy.rawdata"]
Do[PutAppend[N[d[[i,j,k]],16],"dxy.rawdata",{k,3},{j,15},{i,4}]
```

```

(* A Mathematica script to generate the U matrix,
   the spatial dependence array needed in the anisotropic scattering
   calculation. X-Y Geometry *)
<<f.vol.xy.dat
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}]
s = {x, y}
u = Table[0, {i, 15}, {j, 15}, {l, 2}]
Do[
  Do[u[[i, j, l]] = VolInt[f[[j]]*D[f[[i]], s[[l]]]];
  Print[i, " ", j, " ", l, " ", u[[i, j, l]]],
  {i, 15}, {j, 15}
],
{1, 2}
]
(* Save["w.y.dat", u] *)
Put[U, "uxy.rawdata"]
Do[PutAppend[N[u[[i, j, l]], 16], "uxy.rawdata"], {1, 2}, {j, 15}, {i, 15}]

```

(* A Mathematica script to generate orthonormal trial functions over an hex node

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as 1/sqrt(volume) (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \text{sum}(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\text{sqrt}(\langle g[n],g[n] \rangle - \text{sum}(\langle f[i],g[n] \rangle^2:i=1,n-1)) \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n] \rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

*****)

```
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
Simplify[
Integrate[f,{x,-b/3,0},{y,-x/Sqrt[3]-1/b,x/Sqrt[3]+1/b}]+
Integrate[f,{x,0,b/3},{y,x/Sqrt[3]-1/b,-x/Sqrt[3]+1/b}]]
g = {1,
      x,      y,
      x^2,    x*y,    y^2,
      x^3,    x^2*y,  x*y^2,    y^3,
      x^4,    x^3*y,  x^2*y^2,  x*y^3,    y^4,
      x^5,    x^4*y,  x^3*y^2,  x^2*y^3,  x*y^4,    y^5,
      x^6,    x^5*y,  x^4*y^2,  x^3*y^3,  x^2*y^4,  x*y^5,    y^6}
f = Table[0,{i,28}]
a = Table[0,{i,28}]
f[[1]] = 1
Do[
Print["Generating trial function ",n];
Do[ a[[j]] = VolInt[f[[j]]*g[[n]]],{j,1,n-1} ];
a[[n]] = VolInt[g[[n]]*g[[n]]];
Print["End of VolInt"];
sum = 0;
Do[ sum = sum+a[[j]]^2,{j,n-1} ];
a[[n]] = 1/Sqrt[a[[n]]-sum];
Do[ a[[j]] = -a[[n]]*a[[j]], {j,n-1} ];
sum = 0;
Do[ sum = sum+a[[j]]*f[[j]],{j,n-1} ];
f[[n]] = sum + a[[n]]*g[[n]],
{n,2,28} ]
Save["f.vol.hex.dat",f]
```



```

(* A Mathematica script to generate orthonormal trial functions
over an hexagon side *)
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
              b*Integrate[f, {x, -1/(2b), 1/(2b)}])
g = {1, x, x^2}
l = Table[0, {i, 3}]
a = Table[0, {i, 3}]
l[[1]] = 1
Do[
  Print["Generating trial function ", n];
  Do[ a[[j]] = VolInt[l[[j]]*g[[n]], {j, 1, n-1} ];
  a[[n]] = VolInt[g[[n]]*g[[n]];
  Print["End of VolInt"];
  sum = 0;
  Do[ sum = sum+a[[j]]^2, {j, n-1} ];
  a[[n]] = 1/Sqrt[a[[n]]-sum];
  Do[ a[[j]] = -a[[n]]*a[[j]], {j, n-1} ];
  sum = 0;
  Do[ sum = sum+a[[j]]*l[[j]], {j, n-1} ];
  l[[n]] = sum + a[[n]]*g[[n]],
  {n, 2, 3} ]
Save["f.surf.hex.dat", l]

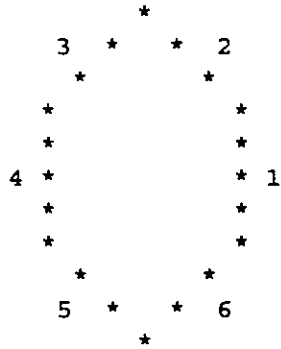
```

```

(* A Mathematica script to generate the P matrix, which takes into account
the spatial dependence of the node interior.
hex Geometry *)
<<f.vol.hex.dat
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
               Simplify[
                 Integrate[f, {x, -b/3, 0}, {y, -x/Sqrt[3]-1/b, x/Sqrt[3]+1/b}] +
                 Integrate[f, {x, 0, b/3}, {y, x/Sqrt[3]-1/b, -x/Sqrt[3]+1/b}]]
s = {x, y}
p = Table[0, {i, 28}, {j, 28}, {k, 2}, {l, 2}]
Do[
  If[k==1,
    Do[p[[{i, j, k, l}] = VolInt[D[f[[{i}], s[[k]]]*D[f[[{j}], s[[l]]]]];
       Print[i, " ", j, " ", k, " ", l, " ", p[[{i, j, k, l}]]];
       p[[{j, i, k, l}] = p[[{i, j, k, l}],
         {i, 28}, {j, i, 28}
       ],
    Do[p[[{i, j, k, l}] = VolInt[D[f[[{i}], s[[k]]]*D[f[[{j}], s[[l]]]]];
       Print[i, " ", j, " ", k, " ", l, " ", p[[{i, j, k, l}]]];
       p[[{j, i, l, k}] = p[[{i, j, k, l}],
         {i, 28}, {j, 28}
       ]
  ],
  {k, 2}, {l, k, 2}
]
(* Save["phex.dat", p] *)
Put[P, "phex.rawdata"]
Do[PutAppend[N[p[[{i, j, k, l}], 16], "phex.rawdata"], {1, 2}, {k, 2}, {j, 28}, {i, 28}]

```

```
(* A Mathematica script to generate the D matrix, which couples
the spatial dependence of the node surfaces to the node interior.
hex Geometry
```



```
*)
```

```
(* Load vector of basis functions *)
```

```
<<f.vol.hex.dat
```

```
(* Load vector of surface trial functions)
```

```
<<f.surf.hex.dat
```

```
(* Define vectors fn where fn is the vector of interior trial functions
transformed to the surface n's coordinate system *)
```

```
b = 3^(3/4)/Sqrt[2]
```

```
(* the notation a /. {x -> x', y -> y'} can be read as transform the
expression "a" replacing x with x' and y with y' *)
```

```
l = l /. x->s
f1 = f /. { x -> b/3, y -> s }
f2 = f /. { x -> b/6(2b s + 1), y -> -1/(4b)(2b s - 3) }
f3 = f /. { x -> b/6(2b s - 1), y -> 1/(4b)(2b s + 3) }
f4 = f /. { x -> -b/3, y -> s }
f5 = f /. { x -> b/6(2b s - 1), y -> -1/(4b)(2b s + 3) }
f6 = f /. { x -> b/6(2b s + 1), y -> 1/(4b)(2b s - 3) }
```

```
(* Define the surface integral on a node surface *)
```

```
SurfInt[f_] := b*Integrate[f, {s, -1/(2b), 1/(2b)}]
```

```
(* Define and initialize the elements of the D matrix *)
```

```
d = Table[0, {i, 6}, {j, 28}, {k, 3}]
```

```
(* Calculate the elements of the D matrix. The D matrix is defined as
D[i, j, k] = SurfInt[f[j]*l[k]] on the ith nodal surface where f and l
are vectors of orthogonal trial functions on the node interior and
nodal surface i *)
```

```
Do[ d[[1, j, k]] = SurfInt[f1[[j]]*l[[k]]];
d[[2, j, k]] = SurfInt[f2[[j]]*l[[k]]];
d[[3, j, k]] = SurfInt[f3[[j]]*l[[k]]];
d[[4, j, k]] = SurfInt[f4[[j]]*l[[k]]];
d[[5, j, k]] = SurfInt[f5[[j]]*l[[k]]];
```

```

    d[[6,j,k]] = SurfInt[f6[[j]]*l[[k]],
      {j,28},{k,3}
  ]

(* Store the completed D matrix *)

(* Save["dhex.dat",d] *)

(* Write the numerical values of the D matrix to an ascii file *)

Put[D,"dhex.rawdata"]
Do[PutAppend[N[d[[i,j,k]],16],"dhex.rawdata",{k,3},{j,28},{i,6}]

```

```

(* A Mathematica script to generate the U matrix,
   the spatial dependence array needed in the anisotropic scattering
   calculation. hex Geometry *)
<<f.vol.hex.dat
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
               Simplify[
                 Integrate[f, {x, -b/3, 0}, {y, -x/Sqrt[3]-1/b, x/Sqrt[3]+1/b}] +
                 Integrate[f, {x, 0, b/3}, {y, x/Sqrt[3]-1/b, -x/Sqrt[3]+1/b}])
s = {x, y}
u = Table[0, {i, 28}, {j, 28}, {1, 2}]
Do[
  Do[u[[i, j, 1]] = VolInt[f[[j]]*D[f[[i]], s[[1]]]];
  Print[i, " ", j, " ", 1, " ", u[[i, j, 1]]],
    {i, 28}, {j, 28}
  ],
  {1, 2}
]
(* Save["uhex.dat", u] *)
Put[U, "uhex.rawdata"]
Do[PutAppend[N[u[[i, j, 1]], 16], "uhex.rawdata"], {1, 2}, {j, 28}, {i, 28}]

```

(* A Mathematica script to generate orthonormal trial functions
over an XYZ node

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as $1/\sqrt{\text{volume}}$ (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \text{sum}(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\sqrt{\langle g[n],g[n]\rangle - \text{sum}(\langle f[i],g[n]\rangle^2:i=1,n-1)} \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n]\rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

*****)

VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}, {z, -1/2, 1/2}]

g = {1,
x,
y,
z,
x^2,
x*y,
y^2,
y*z,
z^2,
z*x,
x^3,
x^2*y,
x*y^2,
y^3,
y^2*z,
y*z^2,
z^3,
z^2*x,
z*x^2,
x*y*z,
x^4,
x^3*y,
x^2*y^2,
x*y^3,
y^4,
y^3*z,
y^2*z^2,
y*z^3,
z^4,
z^3*x,
z^2*x^2,
z*x^3,
x^2*y*z,
x*y^2*z,

```

x*y*z^2}

f = Table[0, {i, 35}]
a = Table[0, {i, 35}]
f[[1]] = 1
Do[
  Print["Generating trial function ", n];
  Do[ a[[j]] = VolInt[f[[j]]*g[[n]], {j, 1, n-1} ];
  a[[n]] = VolInt[g[[n]]*g[[n]]];
  Print["End of VolInt"];
  sum = 0;
  Do[ sum = sum+a[[j]]^2, {j, n-1} ];
  a[[n]] = 1/Sqrt[a[[n]]-sum];
  Do[ a[[j]] = -a[[n]]*a[[j]], {j, n-1} ];
  sum = 0;
  Do[ sum = sum+a[[j]]*f[[j]], {j, n-1} ];
  f[[n]] = sum + a[[n]]*g[[n]],
  {n, 2, 35} ]
Save["f.vol.xyz.dat", f]

```

```
(* A Mathematica script to generate orthonormal trial functions
over an X-Y surface of a X-Y-Z node.
```

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as $1/\sqrt{\text{volume}}$ (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \sum(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\sqrt{\langle g[n],g[n]\rangle - \sum(\langle f[i],g[n]\rangle^2:i=1,n-1)} \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n]\rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

```
*****)
```

```
(* Define the volume integral over the node *)
```

```
VolInt[f_] := Integrate[f,{x,-1/2,1/2},{y,-1/2,1/2}]
```

```
(* Define a vector consisting of the functions making up a complete
second order polynomial *)
```

```
g = {1,x,y,x^2,x*y,y^2}
```

```
(* Define and initialize a vector for the orthogonal trial functions *)
```

```
f = Table[0,{i,6}]
```

```
(* Define and initialize a vector for the trial function coefficients *)
```

```
a = Table[0,{i,6}]
```

```
(* Define the first trial function as 1 *)
```

```
f[[1]] = 1
```

```
(* Begin loop to determine trial functions *)
```

```
Do[
```

```
Print["Generating trial function ",n];
```

```
(* Calculate inner products *)
```

```
Do[ a[[j]] = VolInt[f[[j]]*g[[n]],{j,1,n-1} ];
a[[n]] = VolInt[g[[n]]*g[[n]]];
```

```
(* Calculate sum of squares of inner products *)
```



```

sum = 0;
Do[ sum = sum+a[[j]]^2,{j,n-1} ];

(* Calculate the values of the coefficients *)

a[[n]] = 1/Sqrt[a[[n]]-sum];
Do[ a[[j]] = -a[[n]]*a[[j]], {j,n-1} ];

(* Store the trial function in f[n] *)

sum = 0;
Do[ sum = sum+a[[j]]*f[[j]],{j,n-1} ];
f[[n]] = sum + a[[n]]*g[[n]],

(* End of Do loop *)
{n,2,6} ]

(* Save the set of trial functions *)

Save["f.surfx.yz.dat",f]

```

```

(* A Mathematica script to generate the P matrix, which takes into account
the spatial dependence of the node interior.
X-Y-Z Geometry *)
<<f.vol.xyz.dat
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}, {z, -1/2, 1/2}]
s = {x, y, z}
p = Table[0, {i, 35}, {j, 35}, {k, 3}, {l, 3}]
Do[
  If[k==1,
    Do[p[[i, j, k, l]] = VolInt[D[f[[i]], s[[k]]]*D[f[[j]], s[[l]]]];
      Print[i, " ", j, " ", k, " ", l, " ", p[[i, j, k, l]]];
      p[[j, i, k, l]] = p[[i, j, k, l]],
      {i, 35}, {j, i, 35}
    ],
    Do[p[[i, j, k, l]] = VolInt[D[f[[i]], s[[k]]]*D[f[[j]], s[[l]]]];
      Print[i, " ", j, " ", k, " ", l, " ", p[[i, j, k, l]]];
      p[[j, i, l, k]] = p[[i, j, k, l]],
      {i, 35}, {j, 35}
    ]
  ],
  {k, 3}, {l, k, 3}
]
(* Save["pxyz.dat", p] *)
Put [P, "pxyz.rawdata"]
Do [PutAppend [N[p[[i, j, k, l]], 16], "pxyz.rawdata"], {l, 3}, {k, 3}, {j, 35}, {i, 35}]

```

```

(* A Mathematica script to generate the D matrix, which couples
the spatial dependence of the node surfaces to the node interior.
X-Y-Z Geometry *)
<<f.surfx.yz.dat
l = f /. {x->t,y->s}
Clear[f]
<<f.vol.xyz.dat
f1 = f /. {x->1/2,y->t,z->s}
f2 = f /. {y->1/2,x->t,z->s}
f3 = f /. {z->1/2,x->t,y->s}
f4 = f /. {x->-1/2,y->t,z->s}
f5 = f /. {y->-1/2,x->t,z->s}
f6 = f /. {z->-1/2,x->t,y->s}
(*mx = Table[1,{i,35}]
my = Table[1,{i,35}]
mz = Table[1,{i,35}]
mxm = {2,6,10,11,13,18,20,22,24,30,32,34,35}
mym = {3,6,8,12,14,16,20,22,24,26,28,33,35}
mzm = {5,8,10,15,17,19,20,26,28,29,32,33,34}
Do[ mx[[mxm[[i]]]] = -1;
    my[[mym[[i]]]] = -1;
    mz[[mzm[[i]]]] = -1,
    {i,13}
] *)
SurfInt[f_] := Integrate[f,{s,-1/2,1/2},{t,-1/2,1/2}]
d = Table[0,{i,6},{j,35},{k,6}]
Do[ d[[1,j,k]] = SurfInt[f1[[j]]*1[[k]];
    Print[" side 1 ",d[[1,j,k]];
    (*d[[3,j,k]] = mx[[j]]*d[[2,j,k]];*)
    d[[3,j,k]] = SurfInt[f4[[j]]*1[[k]];
    Print[" side 3 ",d[[3,j,k]];
    d[[2,j,k]] = SurfInt[f2[[j]]*1[[k]];
    Print[" side 2 ",d[[2,j,k]];
    (*d[[4,j,k]] = my[[j]]*d[[2,j,k]];*)
    d[[4,j,k]] = SurfInt[f5[[j]]*1[[k]];
    Print[" side 4 ",d[[4,j,k]];
    d[[5,j,k]] = SurfInt[f3[[j]]*1[[k]];
    Print[" side 5 ",d[[5,j,k]];
    (*d[[6,j,k]] = mz[[j]]*d[[5,j,k]];*)
    d[[6,j,k]] = SurfInt[f6[[j]]*1[[k]];
    Print[" side 6 ",d[[6,j,k]],
    {j,35},{k,6}
]
(* Save["dxyz.dat",d] *)
Put[D,"dxyz.rawdata"]
Do[PutAppend[N[d[[i,j,k]],16],"dxyz.rawdata"],{k,6},{j,35},{i,6}]

```

```

(* A Mathematica script to generate the U matrix,
   the spatial dependence array needed in the anisotropic scattering
   calculation. X-Y-Z Geometry *)
<<f.vol.xyz.dat
VolInt[f_] := Integrate[f, {x, -1/2, 1/2}, {y, -1/2, 1/2}, {z, -1/2, 1/2}]
s = {x, y, z}
u = Table[0, {i, 35}, {j, 35}, {l, 3}]
Do[
  Do[u[[i, j, l]] = VolInt[f[[j]]*D[f[[i]], s[[l]]]];
    Print[i, " ", j, " ", l, " ", u[[i, j, l]]],
      {i, 35}, {j, 35}
    ],
  {l, 3}
]
(* Save["uxyz.dat", u] *)
Put[U, "uxyz.rawdata"]
Do[PutAppend[N[u[[i, j, l]], 16], "uxyz.rawdata"], {l, 3}, {j, 35}, {i, 35}]

```

(* A Mathematica script to generate orthonormal trial functions over an hex-Z node. Complete expansion order is: 6th order in X and Y, 4th order in Z.

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as $1/\sqrt{\text{volume}}$ (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \sum(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\sqrt{\langle g[n],g[n] \rangle - \sum(\langle f[i],g[n] \rangle^2:i=1,n-1)} \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n] \rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

*****)

```
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
Simplify[
Integrate[f,{x,-b/3,0},{y,-x/Sqrt[3]-1/b,x/Sqrt[3]+1/b},{z,-1/2,1/2}]+
Integrate[f,{x,0,b/3},{y,x/Sqrt[3]-1/b,-x/Sqrt[3]+1/b},{z,-1/2,1/2}]]
g = {1,
x,y,z,
x^2,
x*y,
y^2,
y*z,
z^2,
z*x,
x^3,
x^2*y,
x*y^2,
y^3,
y^2*z,
y*z^2,
z^3,
z^2*x,
z*x^2,
x*y*z,
x^4,
x^3*y,
x^2*y^2,
x*y^3,
y^4,
y^3*z,
y^2*z^2,
y*z^3,
z^4,
z^3*x,
z^2*x^2,
z*x^3,
```

```

    x^2*y*z,
    x*y^2*z,
    x*y*z^2,
x^5,
    x^4*y,
    x^3*y^2,
    x^2*y^3,
    x*y^4,
    y^5,
x^6,
    x^5*y,
    x^4*y^2,
    x^3*y^3,
    x^2*y^4,
    x*y^5,
    y^6
}

```

```

f = Table[0, {i, 48}]
a = Table[0, {i, 48}]
f[[1]] = 1
Do[
  Print["Generating trial function ", n];
  Do[ a[[j]] = VolInt[f[[j]]*g[[n]], {j, 1, n-1} ];
  a[[n]] = VolInt[g[[n]]*g[[n]];
  Print["End of VolInt"];
  sum = 0;
  Do[ sum = sum+a[[j]]^2, {j, n-1} ];
  a[[n]] = 1/Sqrt[a[[n]]-sum];
  Do[ a[[j]] = -a[[n]]*a[[j]], {j, n-1} ];
  sum = 0;
  Do[ sum = sum+a[[j]]*f[[j]], {j, n-1} ];
  f[[n]] = sum + a[[n]]*g[[n]],
  {n, 2, 48} ]
Save["f.hex3d64.dat", f]

```

```
(* A Mathematica script to generate orthonormal trial functions
over an XY surface FOR A SIDE OF A HEX CAN
```

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as $1/\sqrt{\text{volume}}$ (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \sum(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\sqrt{\langle g[n],g[n]\rangle - \sum(\langle f[i],g[n]\rangle^2:i=1,n-1)} \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n]\rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

```
*****)
```

```
(* Define the volume integral over the node *)
```

```
VolInt[f_] := (b=3^(3/4)/Sqrt[2] ;
b*Integrate[f,{x,-1/(2 b),1/(2 b)},{y,-1/2,1/2}])
```

```
(* Define a vector consisting of the functions making up a complete
fourth order polynomial *)
```

```
g = {1,x,y,x^2,x*y,y^2}
```

```
(* Define and initialize a vector for the orthogonal trial functions *)
```

```
f = Table[0,{i,6}]
```

```
(* Define and initialize a vector for the trial function coefficients *)
```

```
a = Table[0,{i,6}]
```

```
(* Define the first trial function as 1 *)
```

```
f[[1]] = 1
```

```
(* Begin loop to determine trial functions *)
```

```
Do[
```

```
Print["Generating trial function ",n];
```

```
(* Calculate inner products *)
```

```
Do[ a[[j]] = VolInt[f[[j]]*g[[n]]],{j,1,n-1} ];
a[[n]] = VolInt[g[[n]]*g[[n]]];
```

```
(* Calculate sum of squares of inner products *)
```

```

sum = 0;
Do[ sum = sum+a[[j]]^2,{j,n-1} ];

(* Calculate the values of the coefficients *)

a[[n]] = 1/Sqrt[a[[n]]-sum];
Do[ a[[j]] = -a[[n]]*a[[j]], {j,n-1} ];

(* Store the trial function in f[n] *)

sum = 0;
Do[ sum = sum+a[[j]]*f[[j]],{j,n-1} ];
f[[n]] = sum + a[[n]]*g[[n]],

(* End of Do loop *)
{n,2,6} ]

(* Save the set of trial functions *)

Save["f.surfxy.hexz.dat",f]

```


(* A Mathematica script to generate orthonormal trial functions
over an hex surface FOR A SIDE OF A HEX CAN

- (1) Define volume integral over the domain
- (2) Define vector of complete polynomials up to order desired (g)
- (3) Define first orthonormal trial function as 1/sqrt(volume) (f[1])
- (4) Loop over lin. indep. polynomials to det. trial function coefficients

$$f[n] = \text{sum}(a[i]f[i]:i=1,n-1)+a[n]g[n] \quad \{1\}$$

- (a) calculate inner products and store in a
- (b) using inner products calc a[n] -

$$a[n] = 1/\text{sqrt}(\langle g[n],g[n]\rangle - \text{sum}(\langle f[i],g[n]\rangle^2:i=1,n-1)) \quad \{2\}$$

- (c) using a[n], calculate all a[i]

$$a[i] = -a[n]\langle f[i],g[n]\rangle, \quad i=1,n-1 \quad \{3\}$$

- (d) using the coefficients stored in a, calculate the nth trial function using {1}

*****)

```
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
Simplify[
Integrate[f, {x, -b/3, 0}, {y, -x/Sqrt[3]-1/b, x/Sqrt[3]+1/b}] +
Integrate[f, {x, 0, b/3}, {y, x/Sqrt[3]-1/b, -x/Sqrt[3]+1/b}]]
g = {1,
      x, y,
      x^2, x*y, y^2}
f = Table[0, {i, 6}]
a = Table[0, {i, 6}]
f[[1]] = 1
Do[
Print["Generating trial function ", n];
Do[ a[[j]] = VolInt[f[[j]]*g[[n]]], {j, 1, n-1} ];
a[[n]] = VolInt[g[[n]]*g[[n]]];
Print["End of VolInt"];
sum = 0;
Do[ sum = sum+a[[j]]^2, {j, n-1} ];
a[[n]] = 1/Sqrt[a[[n]]-sum];
Do[ a[[j]] = -a[[n]]*a[[j]], {j, n-1} ];
sum = 0;
Do[ sum = sum+a[[j]]*f[[j]], {j, n-1} ];
f[[n]] = sum + a[[n]]*g[[n]],
{n, 2, 6} ]
Save["f.surfh.hexz.dat", f]
```

```

(* A Mathematica script to generate the P matrix, which takes into account
the spatial dependence of the node interior.
hex-Z Geometry *)
<<f64.vol.hexz.dat
f = Simplify[N[f,15]]
VolInt[f_] := (
  b = 3^(3/4)/Sqrt[2];
  Simplify[
    Integrate[f, {x, -b/3, 0}, {y, -x/Sqrt[3]-1/b, x/Sqrt[3]+1/b}, {z, -1/2, 1/2}] +
    Integrate[f, {x, 0, b/3}, {y, x/Sqrt[3]-1/b, -x/Sqrt[3]+1/b}, {z, -1/2, 1/2}]]
s = {x, y, z}
p = Table[0, {i, 48}, {j, 48}, {k, 3}, {l, 3}]
Do[
  If[k==1,
    Do[
      p[[i,j,k,l]] = VolInt[D[f[[i]],s[[k]]]*D[f[[j]],s[[l]]]];
      Print[ i, " ", j, " ", k, " ", l, " ", N[ p[[i,j,k,l]] ] ];
      p[[j,i,k,l]] = p[[i,j,k,l]],
      {i, 48}, {j, i, 48}
    ],
    Do[
      p[[i,j,k,l]] = VolInt[D[f[[i]],s[[k]]]*D[f[[j]],s[[l]]]];
      Print[ i, " ", j, " ", k, " ", l, " ", N[ p[[i,j,k,l]] ] ];
      p[[j,i,l,k]] = p[[i,j,k,l]],
      {i, 48}, {j, 48}
    ]
  ];
  (* Save["s64.phexz.dat",p],
    {k,3}, {l,k,3}
  *)
stmp = OpenWrite["s64.phexz.rawdata"]
WriteString[stmp,"P \n"]
Do[
  Write[ stmp, Chop[ N[ p[[i,j,k,l]], 12 ], 10^-12 ] ] ,
  {l,3},{k,3},{j,48},{i,48}
]
Close[stmp]

```

(* Script for calculating the hex-z D matrix. The sides are numbered as follows:

Sides 1-6 : rectangular faces starting at $x = b/3$, numbered counter clockwise

Side 7 : upper hexagonal face

Side 8 : lower hexagonal face

*)

(* Number of internal moments *)

(* 4th order in z, 6th order in xy *)

nFluxMom = 48

(* Number of surface moments *)

(* quadratic in s and t *)

nCurrMom = 6

(* Load vector of surface trial functions *)

(* xy trial functions on the rectangular faces *)

<< f.surfxy.hexz.dat

f=Simplify[N[f,15]]

hxy = f

hxy = hxy /. { x -> s, y -> t }

Clear[f]

(* hex trial functions on the z faces *)

<< f.surfh.hexz.dat

f=Simplify[N[f,15]]

hz = f

hz = hz /. { x -> s, y -> t }

Clear[f]

(* define a function which returns the appropriate surface trial function *)

h[i_,j_] = (If[i < 7, hxy[[j]], hz[[j]]]);

(* Define surface integral: sides 1-6 - rectangular domain
sides 7-8 - hexagonal domain *)

surfInt[f_,i_] = (

If[

i < 7, b=3^(3/4)/Sqrt[2];

b*Integrate[f, {s, -1/(2 b), 1/(2 b)}, {t, -1/2, 1/2}],

b = 3^(3/4)/Sqrt[2];

Simplify[

Integrate[f, {s, -b/3, 0}, {t, -s/Sqrt[3]-1/b, s/Sqrt[3]+1/b}] +

Integrate[f, {s, 0, b/3}, {t, s/Sqrt[3]-1/b, -s/Sqrt[3]+1/b}]

]

]

);

(* Load vector of internal basis functions, f *)

<<f64.vol.hexz.dat

f=Simplify[N[f,15]]

(* Define vectors fp where fp[[n]] is the vector of interior trial functions

transformed to the surface n's coordinate system *)

```
b = 3^(3/4)/Sqrt[2]
```

```
fp = Table[ 0, {i,8} ]
fp[[1]] = f /. { x -> b/3, y -> s, z -> t }
fp[[2]] = f /. { x -> b/6(2b s + 1), y -> -1/(4b) (2b s - 3), z -> t }
fp[[3]] = f /. { x -> b/6(2b s - 1), y -> 1/(4b) (2b s + 3), z -> t }
fp[[4]] = f /. { x -> -b/3, y -> s, z -> t }
fp[[5]] = f /. { x -> b/6(2b s - 1), y -> -1/(4b) (2b s + 3), z -> t }
fp[[6]] = f /. { x -> b/6(2b s + 1), y -> 1/(4b) (2b s - 3), z -> t }
fp[[7]] = f /. { z -> 1/2, x -> s, y -> t }
fp[[8]] = f /. { z -> -1/2, x -> s, y -> t }
```

(* Define and initialize the elements of the D matrix *)

```
d = Table[0, {i,8}, {j,nFluxMom}, {k,nCurrMom}]
```

```
Do[
  d[[{i,j,k}] = surfInt[ Simplify[ N[ fp[[{i,j}] h[i,k], 14 ] ], i ];
  Print["Surface ",i," Pair[ ",j," ",k," ] = ", N[ d[[{i,j,k}] ] ],
    {i,8}, {j,nFluxMom}, {k,nCurrMom}
]
```

(* Store the completed D matrix *)

(* Save["s64.dhexz.dat",d] *)

(* Write the numerical values of the D matrix to an ascii file *)

```
stmp = OpenWrite["s64.dhexz.rawdata"]
WriteString[stmp,"D \n"]
Do[
  Write[ stmp, Chop[ N[ d[[{i,j,k}],12 ], 10^-12 ] ],
    {k,nCurrMom}, {j,nFluxMom}, {i,8}
]
Close[stmp]
```

```

(* A Mathematica script to generate the U matrix,
the spatial dependence array needed in the anisotropic scattering
calculation. hex-Z Geometry *)
<<f64.vol.hexz.dat
f=Simplify[N[f,15]]
VolInt[f_] := (b = 3^(3/4)/Sqrt[2];
Simplify[
Integrate[f, {x, -b/3, 0}, {y, -x/Sqrt[3]-1/b, x/Sqrt[3]+1/b}, {z, -1/2, 1/2}]+
Integrate[f, {x, 0, b/3}, {y, x/Sqrt[3]-1/b, -x/Sqrt[3]+1/b}, {z, -1/2, 1/2}]]
s = {x, y, z}
u = Table[0, {i, 48}, {j, 48}, {l, 3}]
Do[
Do[u[[i, j, l]] = VolInt[f[[j]]*D[f[[i]], s[[l]]]];
Print[i, " ", j, " ", l, " ", u[[i, j, l]]],
{i, 48}, {j, 48}
],
{l, 3}
]
(* Save["s64.uhexz.dat", u] *)
Put[U, "s64.uhexz.rawdata"]
Do[PutAppend[N[u[[i, j, l]], 16], "s64.uhexz.rawdata"], {l, 3}, {j, 48}, {i, 48}]

```

```

(* A Mathematica script to generate the P matrix, which takes into account
the angular dependence of the node interior.
2d Geometry *)
ifa=9
rules = {x -> Cos[th],y -> Sin[th]Cos[ph],z -> Sin[th]Sin[ph]}
<<even_parity.functions.2d
g = g /. rules
o={Cos[th],Sin[th]*Cos[ph]}
<<AngInt.math
h = Table[0,{i,2},{j,2},{k,ifa},{l,ifa}]
Do[a=AngInt[ o[[i]] o[[j]] g[[k]] g[[l]] ];
  h[[i,j,k,l]]=a;
  a=a, (*
  h[[i,j,l,k]]=a;
  h[[j,i,k,l]]=a;
  h[[j,i,l,k]]=a, *)
  {i,2},{j,2},{k,ifa},{l,ifa}]
(* Save["hxy.dat",h] *)
stmp = OpenWrite["hxy.rawdata"]
WriteString[stmp,"H \n"]
Do[Write[stmp,N[h[[i,j,k,l]],16]],{l,ifa},{k,ifa},{j,2},{i,2}]
Close[stmp]

```

```

(* A Mathematica script to generate the P matrix, which takes into account
the angular dependence of the node interior.
3d Geometry *)
rules = {x -> Cos[th], y -> Sin[th]Cos[ph], z -> Sin[th]Sin[ph]}
<<even_parity.functions.3d
<<AngInt.math
g = g /. rules
o={Cos[th], Sin[th]*Cos[ph], Sin[th] Sin[ph]}
h = Table[0, {i, 3}, {j, 3}, {k, 15}, {l, 15}]
Do[
  a=AngInt[ o[[i]] o[[j]] g[[k]] g[[l]] ];
  h[[i, j, k, l]]=a;
  h[[i, j, l, k]]=a;
  h[[j, i, k, l]]=a;
  h[[j, i, l, k]]=a,
  {i, 3}, {j, i, 3}, {k, 15}, {l, k, 15}]
(* Save["hxyz.dat", h] *)
stmp = OpenWrite["hxyz.rawdata"]
WriteString[stmp, "H \n"]
Do[Write[stmp, N[h[[i, j, k, l]], 16]], {l, 15}, {k, 15}, {j, 3}, {i, 3}]
Close[stmp]

```

APPENDIX B

DESCRIPTION OF FILE COMPTS

```

C*****
C
C          PREPARED 3/7/78 AT ANL
C          LAST REVISED 08/31/95
C
CF          COMPTS
CE          MACROSCOPIC COMPOSITION CROSS SECTIONS
C
C*****
    
```

```

C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE                PRESENT IF
CS          =====
CS          SPECIFICATIONS              ALWAYS
CS          COMPOSITION INDEPENDENT DATA  ALWAYS
CS          ***** (REPEAT FOR ALL COMPOSITIONS)
CS          *          COMPOSITION SPECIFICATIONS          ALWAYS
CS          *          ***** (REPEAT FOR ALL ENERGY GROUPS
CS          * *          IN THE ORDER OF DECREASING
CS          * *          ENERGY)
CS          * *          COMPOSITION MACROSCOPIC GROUP      ALWAYS
CS          * *          CROSS SECTIONS
CS          *****
CS          POWER CONVERSION FACTORS          ALWAYS
CS
C
C-----
    
```

```

CD          NGROUP          NUMBER OF ENERGY GROUPS.
CD          ICHI            PROMPT FISSION SPECTRUM FLAG FOR THIS
CD                        COMPOSITION. ICHI=-1 IF COMPOSITION USES THE
CD                        SET-WIDE PROMPT CHI GIVEN IN SET CHI RECORD
CD                        (BELOW). ICHI=0 IF COMPOSITION IS NOT
CD                        FISSIONABLE. ICHI=1 FOR COMPOSITION PROMPT CHI
CD                        VECTOR. ICHI=NGROUP FOR COMPOSITION PROMPT CHI
CD                        MATRIX.
CD          NUP(I)         NUMBER OF GROUPS OF UPSCATTERING INTO GROUP I
CD                        FROM LOWER ENERGY GROUPS FOR THE CURRENT
CD                        COMPOSITION
CD          NDN(I)         NUMBER OF GROUPS OF DOWNSCATTERING INTO GROUP I
CD                        FROM HIGHER ENERGY GROUPS FOR THE CURRENT
CD                        COMPOSITION
CD          ISCHI          PROMPT FISSION SPECTRUM FLAG. ISCHI=0 IF
CD                        THERE IS NO SET-WIDE PROMPT CHI. ISCHI=1 IF
CD                        THERE IS A SET-WIDE PROMPT CHI VECTOR.
CD                        ISCHI=NGROUP IF THERE IS A SET-WIDE PROMPT
CD                        CHI MATRIX.
CD          NFAM           NUMBER OF DELAYED NEUTRON FAMILIES.
CD          MULT           2 FOR IBM MACHINES, 1 OTHERWISE.
    
```



```

C-----
CR          SPECIFICATIONS (TYPE 1)
C
CL  NCMP , NGROUP , ISCHI , NFCMP , MAXUP , MAXDN , NFAM , NDUM1 , NDUM2 , NDUM3
C
CW  10
C
CD  NCMP          NUMBER OF COMPOSITIONS.
CD  NFCMP        NUMBER OF FISSIONABLE COMPOSITIONS.
CD  MAXUP        MAXIMUM NUMBER OF GROUPS OF UPSCATTERING FOR
CD              THE SET.
CD  MAXDN        MAXIMUM NUMBER OF GROUPS OF DOWNSCATTERING
CD              FOR THE SET.
CD  MAXORD       ANISOTROPIC SCATTERING ORDER
CD  NDUM2        RESERVED.
CD  NDUM3        RESERVED.
C
C-----

```

```

C-----
CR          COMPOSITION INDEPENDENT DATA (TYPE 2)
C
CC          ALWAYS PRESENT
C
CL  ((CHI (I , J) , I=1 , ISCHI) , J=1 , NGROUP) , (VEL (J) , J=1 , NGROUP) ,
CL  1 (EMAX (J) , J=1 , NGROUP) , EMIN , ((CHID (J , K) , J=1 , NGROUP) , K=1 , NFAM) ,
CL  2 (FLAM (K) , K=1 , NFAM) , (NKFAM (J) , J=1 , NCMP)
C
CW  MULT* (NGROUP* (ISCHI+2+NFAM) +1+NFAM) +NCMP
C
CD  CHI          PROMPT FISSION FRACTION INTO GROUP J FROM
CD              GROUP I. IF ISCHI=1, THE LIST REDUCES TO
CD              (CHI (J) , J=1 , NGROUP) , WHERE CHI (J) IS THE
CD              FISSION FRACTION INTO GROUP J.
CD  VEL          MEAN NEUTRON VELOCITY IN GROUP J (CM/SEC) .
CD  EMAX         MAXIMUM ENERGY BOUND OF GROUP J (EV) .
CD  EMIN         MINIMUM ENERGY BOUND OF SET (EV) .
CD  CHID         FRACTION OF DELAYED NEUTRONS EMITTED INTO
CD              NEUTRON ENERGY GROUP J FROM PRECURSOR
CD              FAMILY K.
CD  FLAM         DELAYED NEUTRON PRECURSOR DECAY CONSTANT
CD              FOR FAMILY K.
CD  NKFAM        NUMBER OF FAMILIES TO WHICH FISSION IN
CD              COMPOSITION J CONTRIBUTES DELAYED NEUTRON
CD              PRECURSORS.
C
C-----

```

```

C-----
CR          COMPOSITION SPECIFICATIONS (TYPE 3)
C
CC          ALWAYS PRESENT
C
CL  ICHI , (NUP (I) , I=1 , NGROUP) , (NDN (I) , I=1 , NGROUP) ,
CL  1 (NUMFAM (I) , I=1 , NKFAMI)
C
CC  NKFAMI = NKFAM (K)

```

C
 CW 1+2*NGROUP+NKFAMI
 C
 CD NUMFAM FAMILY NUMBER OF THE I-TH YIELD VECTOR IN
 CD ARRAY SNUDEL(I).
 C
 C-----

C-----
 CR COMPOSITION MACROSCOPIC GROUP CROSS SECTIONS (TYPE 4)
 C
 CC ALWAYS PRESENT
 C
 CL XA,XTOT,XREM,XTR,XF,XNF,(CHI(I),I=1,ICHI),
 CL 1(XSCATU(I),I=1,NUMUP),XSCATJ,(XSCATD(I),I=1,NUMDN),
 CL 2PC,A1,B1,A2,B2,A3,B3,(SNUDEL(I),I=1,NKFAMI),XN2N,
 CL 3((XSCAUP(I,L),I=1,NUMUP),(XSCAJP(I,L),I=1,NUMDN),L=1,MAXORD)
 C
 CC NUMUP = NUP FOR THE CURRENT GROUP
 CC NUMDN = NDN FOR THE CURRENT GROUP
 CC NKFAMI = NKFAM(K)
 C
 CW MULT*(15+ICHI+NUMUP+NUMDN+NKFAMI) IF ICHI.GT.0
 CW MULT*(15+NUMUP+NUMDN+NKFAMI) IF ICHI.EQ.-1
 CW MULT*(13+NUMUP+NUMDN+NKFAMI) IF ICHI.EQ.0
 C
 CD XA ABSORPTION CROSS SECTION.
 CD XTOT TOTAL CROSS SECTION.
 CD XREM REMOVAL CROSS SECTION, TOTAL CROSS SECTION
 CD FOR REMOVING A NEUTRON FROM GROUP J DUE TO ALL
 CD PROCESSES.
 CD XTR TRANSPORT CROSS SECTION.
 CD XF FISSION CROSS SECTION, PRESENT ONLY IF
 CD ICHI.NE.0.
 CD XNF TOTAL NUMBER OF NEUTRONS EMITTED PER FISSION
 CD TIMES XF, PRESENT ONLY IF ICHI.NE.0.
 CD CHI PROMPT FISSION FRACTION INTO GROUP J FROM
 CD GROUP I, PRESENT ONLY IF ICHI.GT.0. IF ICHI=1,
 CD THE LIST REDUCES TO THE SINGLE NUMBER CHI,
 CD WHICH IS THE PROMPT FISSION FRACTION INTO
 CD GROUP J.
 CD XSCATU TOTAL SCATTERING CROSS SECTION INTO GROUP J
 CD FROM GROUPS J+NUP(J),J+NUP(J)-1,...,J+2,J+1,
 CD PRESENT ONLY IF NUP(J).GT.0.
 CD XSCATJ TOTAL SELF-SCATTERING CROSS SECTION FROM
 CD GROUP J TO GROUP J.
 CD XSCATD TOTAL SCATTERING CROSS SECTION INTO GROUP J
 CD FROM GROUPS J-1,J-2,...,J-NDN(J), PRESENT
 CD ONLY IF NDN(J).GT.0.
 CD PC PC TIMES THE GROUP J REGION INTEGRATED
 CD FLUX FOR THE REGIONS CONTAINING THE CURRENT
 CD COMPOSITION YIELDS THE POWER IN WATTS IN THOSE
 CD REGIONS AND ENERGY GROUP J DUE TO FISSIONS
 CD AND NON-FISSION ABSORPTIONS.
 CD A1 FIRST DIMENSION DIRECTIONAL DIFFUSION
 CD COEFFICIENT MULTIPLIER.
 CD B1 FIRST DIMENSION DIRECTIONAL DIFFUSION
 CD COEFFICIENT ADDITIVE TERM.
 CD A2 SECOND DIMENSION DIRECTIONAL DIFFUSION

CD COEFFICIENT MULTIPLIER. -
 CD B2 SECOND DIMENSION DIRECTIONAL DIFFUSION -
 CD COEFFICIENT ADDITIVE TERM. -
 CD A3 THIRD DIMENSION DIRECTIONAL DIFFUSION -
 CD COEFFICIENT MULTIPLIER. -
 CD B3 THIRD DIMENSION DIRECTIONAL DIFFUSION -
 CD COEFFICIENT ADDITIVE TERM. -
 CD SNUDEL NUMBER OF DELAYED NEUTRON PRECURSORS PRODUCED -
 CD IN FAMILY NUMBER NUMFAM(I) PER FISSION -
 CD IN GROUP J. -
 CD XN2N N,2N REACTION CROSS SECTION -
 C -
 CN THE MACROSCOPIC XN2N(J) TIMES THE FLUX IN GROUP -
 CN J GIVES THE RATE AT WHICH N,2N REACTIONS OCCUR -
 CN IN GROUP J. THUS, FOR N,2N SCATTERING, -
 CN $XN2N(J) = 0.5 * (\text{SUM OF SCAT}(J \text{ TO } G))$ SUMMED OVER -
 CN ALL G WHERE SCAT IS THE N,2N SCATTERING MATRIX. -
 C -
 CD XSCAUP SAME AS XSCATU BUT FOR ANISOTROPIC ORDER L -
 CD XSCAJP SAME AS XSCATJ BUT FOR ANISOTROPIC ORDER L -
 C -
 C-----

C-----
 CR POWER CONVERSION FACTORS (TYPE 5) -
 C -
 CC ALWAYS PRESENT -
 C -
 CL (FPWS(I), I=1, NCMP), (CPWS(I), I=1, NCMP) -
 C -
 CW 2*MULT*NCMP -
 C -
 CD FPWS FISSIONS/WATT-SECOND FOR EACH COMPOSITION -
 CD CPWS CAPTURES/WATT-SECOND FOR EACH COMPOSITION -
 C -
 CN -
 CN IF ENERGY CONVERSION DATA ARE SUPPLIED FOR -
 CN NEITHER FISSION NOR CAPTURE FOR A PARTICULAR -
 CN COMPOSITION, BOTH FPWS AND CPWS SHOULD BE SET -
 CN TO THE ARTIFICIAL VALUE OF -1.0E+20 FOR THAT -
 CN COMPOSITION. -
 CN -
 CN IF EITHER FPWS(I) OR CPWS(I) (BUT NOT BOTH) IS -
 CN SPECIFIED BY THE USER FOR COMPOSITION I, THEN -
 CN THE ITEM WHICH IS NOT SPECIFIED SHOULD BE SET -
 CN TO ZERO FOR COMPOSITION I. -
 CN -
 CN AT THE PRESENT TIME, REBUS-3 IS THE ONLY CODE -
 CN WHICH USES THE DATA IN RECORD TYPE 5, AND -
 CN THE -1.0E+20 NUMBERS ARE USED TO INDICATE THAT -
 CN THE USER HAS SPECIFIED NEITHER FISSION NOR -
 CN NON-FISSION CAPTURE ENERGY CONVERSION FACTORS -
 CN FOR VARIOUS COMPOSITIONS. -
 C -
 C-----

CEOF

APPENDIX C

DESCRIPTION OF VARIANT OUTPUT FILE NHFLUX

```

C*****
C
C          PREPARED 3/01/82
C          LAST REVISED 8/31/95
C          REVISED 5/29/91 FOR DIF3D 7.0
C
CF          NHFLUX
CE          REGULAR NODAL FLUX-MOMENTS AND INTERFACE PARTIAL CURRENTS
C
CN          ORDER OF GROUPS IS ACCORDING TO DECREASING
CN          ENERGY. NOTE THAT DOUBLE PRECISION FLUXES ARE
CN          GIVEN WHEN MULT=2
C
C*****
    
```

```

C-----
CS          FILE STRUCTURE
CS
CS          RECORD TYPE          RECORD    PRESENT IF
CS          =====          =====
CS          FILE IDENTIFICATION          ALWAYS
CS          SPECIFICATIONS          1D      ALWAYS
CS          INTEGER POINTERS          2D      NSURF.GT.1
CS
CS          ***** (REPEAT FOR ALL GROUPS)
CS          *          FLUX MOMENTS          3D      ALWAYS
CS          *          XY-DIRECTED PARTIAL CURRENTS          4D      ALWAYS
CS          *          Z -DIRECTED PARTIAL CURRENTS          5D      NDIM.EQ.3
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 - NHFLUX - (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,NSURF,      -
CL   NMOM,NINTXY,NPCXY,NSCOEF,ITRORD,IAPRX,ILEAK,IAPRXZ,ILEAKZ,  -
CL   IORDER,IDUM                                               -
C   -
CW   20 =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   ITER                OUTER ITERATION NUMBER AT WHICH FLUX WAS -
CD                       WRITTEN                              -
CD   EFFK                EFFECTIVE MULTIPLICATION FACTOR      -
CD   POWER               POWER IN WATTS TO WHICH FLUX IS NORMALIZED -
CD   NSURF               NUMBER OF XY-PLANE SURFACES PER NODE. -
CD   NMOM                NUMBER OF FLUX MOMENTS IN NODAL APPROXIMATION -
CD   NINTXY              NUMBER OF MESH CELLS (NODES) ON XY-PLANE -
CD   NPCXY               NUMBER OF XY-DIRECTED PARTIAL CURRENTS ON -
CD                       XY-PLANE                             -
CD   NSCOEF              NUMBER OF PARTIAL CURRENT MOMENTS PER NODE -
CD                       SURFACE                              -
CD   ITRORD              ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
CD                       SOURCE WITHIN THE NODE               -
CD   IAPRX               ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
CD                       FLUXES WITHIN THE NODE               -
CD   ILEAK               ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
CD                       LEAKAGES ON THE SURFACES OF THE NODES -
CD   IAPRXZ              ORDER OF THE PN EXPANSION OF THE FLUX -
CD   ILEAKZ              ORDER OF THE PN EXPANSION OF THE LEAKAGE -
CD   IORDER              MESH ORDERING SENTINEL               -
CD                       =0, ORIGINAL NODAL ORDERING PRIOR TO DIF3D 7.0 -
CD                       =1, REVISED NODAL ORDERING, DIF3D 7.0 -
CD   IDUM                RESERVED FOR FUTURE USE              -
C   -
CN   IORDER PERMITS DETECTION OF NHFLUX FILES FROM            -
CN   DIF3D VERSIONS PRECEDING DIF3D 7.0                       -
C   -
C-----
C-----
CR   INTEGER POINTERS    (2D RECORD)                            -
C   -
CC   PRESENT IF NSURF.GT.1                                     -
C   -
CL   (IPCPNT(I,J),I=1,NSURF),J=1,NINTXY),(IPCBDY(I),I=1,NPCBDY), -
CL   (ITRMAP(I),I=1,NINTXY)                                    -
C   -
CW   NSURF*NINTXY + NPCBDY + NINTXY =NUMBER OF WORDS         -
C   -
CD   IPCPNT(I,J)        POINTERS TO INCOMING XY-PLANE PARTIAL CURRENTS. -
CD   IPCBDY(I)          POINTERS TO OUTGOING PARTIAL CURRENTS ON OUTER -
CD                       XY-PLANE BOUNDARY.                   -
CD   ITRMAP(I)          TRANSFORMATION MAP BETWEEN NODAL AND GEODST -
CD                       MESH CELL ORDERINGS.                 -
CD   NPCBDY             = NPCXY - NSURF*NINTXY.               -
C   -
CN   IPCBDY WILL INCLUDE OUTGOING PARTIAL CURRENTS           -

```

```

CN          ON CERTAIN SYMMETRY BOUNDARIES TO AVOID VECTOR -
CN          RECURSION IN DIF3D 7.0 AND LATER VERSIONS. -
C          -
CN          THE NODAL ORDERING IN DIF3D 7.0 AND LATER -
CN          VERSIONS HAS ACTIVE NODES ORDERED BY COLOR, -
CN          FOLLOWED BY INACTIVE NODES. -
C-----

```

```

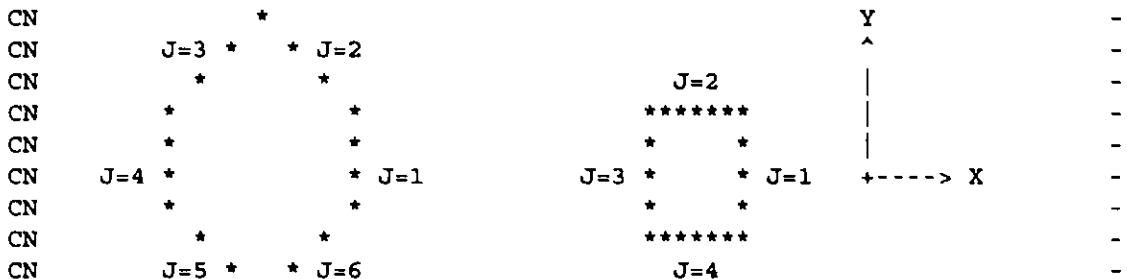
C-----
CR          REGULAR FLUX MOMENTS      (3D RECORD) -
C          -
CL          ((FLUX(I,J),I=1,NMOM),J=1,NINTXY) -----SEE STRUCTURE BELOW----- -
C          -
CW          NMOM*NINTXY*MULT = NUMBER OF WORDS -
C          -
C          DO 1 K=1,NINTK -
C          1 READ(N)  *LIST AS ABOVE* -
C          -
CD          FLUX(I,J)          REGULAR FLUX MOMENTS BY NODE FOR THE PRESENT -
CD          GROUP -
C          -
C-----

```

```

C-----
CR          REGULAR XY-DIRECTED PARTIAL CURRENTS      (4D RECORD) -
C          -
CL          ((PCURRH(I,M),M=1,NSCOEF),I=1,NPCXY) ----SEE STRUCTURE BELOW---- -
C          -
CW          NPCXY*NSCOEF*MULT = NUMBER OF WORDS -
C          -
C          DO 1 K=1,NINTK -
C          1 READ(N)  *LIST AS ABOVE* -
C          -
CD          PCURRH(I,M)          OUTGOING XY-DIRECTED PARTIAL CURRENTS -
CD          ACROSS ALL XY-PLANE SURFACES FOR THE -
CD          THE PRESENT GROUP -
C          -
CN          ELEMENTS I=1,NSURF*NINTXY OF EACH VECTOR PCURRH(.,M) MAP TO -
CN          SURFACE S OF NODE N WHERE  S = MOD(I-1,NSURF)+1  AND -
CN          N = (I-1)/NSURF + 1 -
CN          -
CN          THE REMAINING ELEMENTS (PCURRH(I,M),I=NSURF*NINTXY+1,NPCXY), -
CN          IF ANY, CORRESPOND TO INCOMING PARTIAL CURRENTS (M=1) OR INCOMING- -
CN          HALF-ANGLE INTEGRATED FLUXES (M=2) FOR NODE SURFACES ON THE OUTER- -
CN          (POSSIBLY IRREGULAR) XY-PLANE BOUNDARY. -
CN          -
CN          THE FOLLOWING ORIENTATION IS USED TO DENOTE -
CN          SURFACES J=1,...,NSURF AND NEIGHBORING NODES J=1,...,NSURF: -
C          -

```



```

CN          *
CN
CN          HEXAGONAL NODES          CARTESIAN NODES
CN          NSURF = 6                NSURF = 4
C
C-----
C-----
CR          REGULAR Z-DIRECTED PARTIAL CURRENTS (5D RECORD)
C
CL          ((PCURRZ(I,M,J), I=1, NINTXY), M=1, NSCOEF), J=1, 2)
CL          -----SEE STRUCTURE BELOW-----
C
CW          NINTXY*NSCOEF*2*MULT = NUMBER OF WORDS--
C
C          DO 1 K=1, NINTK1
C          1 READ(N) *LIST AS ABOVE*
C
CC          WITH NINTK1 = NINTK + 1
C
CD          PCURRZ(I,M,J)          REGULAR Z-DIRECTED PARTIAL CURRENTS (M=1) AND
CD          HALF-ANGLE INTEGRATED FLUXES (M=2) IN
CD          PLUS- (J=1) AND MINUS- (J=2) Z DIRECTIONS
CD          ACROSS ALL AXIAL BOUNDARIES FOR THE PRESENT
CD          GROUP
C
CN          E.G. INCOMING PARTIAL CURRENTS FOR NODE I ON
CN          AXIAL MESH INTERVAL K ARE PCURRZ(I,1,1) ON THE
CN          LOWER BOUNDARY (RECORD K) AND PCURRZ(I,1,2) ON
CN          THE UPPER AXIAL BOUNDARY (RECORD K+1).
C
C-----

```

CEOF

APPENDIX D

DESCRIPTION OF THE BCD INPUT FILE A.DIF3D

```

C*****
C
C              REVISED  8/31/95
C
C          A.DIF3D
CE          ONE-, TWO-, AND THREE-DIMENSIONAL DIFFUSION THEORY
CE          MODULE-DEPENDENT BCD INPUT
C
CN          THIS BCD DATASET MAY BE WRITTEN EITHER
CN          IN FREE FORMAT (UNFORM=A.DIF3D) OR
CN          ACCORDING TO THE FORMATS SPECIFIED FOR EACH
CN          CARD TYPE (DATASET=A.DIF3D) .
CN
CN          COLUMNS 1-2 MUST CONTAIN THE CARD TYPE NUMBER.
CN
CN          A BLANK OR ZERO FIELD GIVES THE DEFAULT OPTION
CN          INDICATED.
CN
CN          NON-DEFAULTED DATA ITEMS ON THE A.DIF3D
CN          DATA SET ALWAYS OVERRIDE THE CORRESPONDING
CN          DATA ON THE RESTART DATA SET DIF3D.
C
C*****

```

```

C-----
CR          PROBLEM TITLE (TYPE 01)
C
CL          FORMAT----- (I2, 4X, 11A6)
C
CD          COLUMNS          CONTENTS...IMPLICATIONS, IF ANY
CD          =====
CD          1-2          01
CD
CD          7-72          ANY ALPHANUMERIC CHARACTERS (1 CARD ONLY) .
C
C-----

```

```

C-----
CR          STORAGE AND DUMP SPECIFICATIONS (TYPE 02)
C
CL          FORMAT----- (I2, 4X, 3I6)
C
CD # COLUMNS          CONTENTS...IMPLICATIONS, IF ANY
CD = =====
CD 1  1-2          02
CD
CD 2  7-12          POINTR CONTAINER ARRAY SIZE IN FAST CORE MEMORY (FCM)
CD                   IN REAL*8 WORDS (DEFAULT=10000) .
CD
CD 3  13-18         POINTR CONTAINER ARRAY SIZE IN EXTENDED CORE
CD                   MEMORY (ECM) IN REAL*8 WORDS (DEFAULT=30000) .
CD

```



```

CD 4 19-24  POINTR DEBUGGING EDIT. -
CD          0...NO DEBUGGING PRINTOUT (DEFAULT). -
CD          1...DEBUGGING DUMP PRINTOUT. -
CD          2...DEBUGGING TRACE PRINTOUT. -
CD          3...BOTH DUMP AND TRACE PRINTOUT. -
C -
C-----

```

```

C-----
CR          PROBLEM CONTROL PARAMETERS (TYPE 03) -
C -

```

```

CL          FORMAT----- (I2,4X,11I6) -
C -

```

| CD # | COLUMNS | CONTENTS...IMPLICATIONS, IF ANY |
|-------|---------|--|
| CD = | ===== | ===== |
| CD 1 | 1-2 | 03 |
| CD 2 | 7-12 | PROBLEM TYPE. 0...K-EFFECTIVE PROBLEM (DEFAULT). 1...FIXED SOURCE PROBLEM. |
| CD 3 | 13-18 | SOLUTION TYPE. 0...REAL SOLUTION (DEFAULT). 1...ADJOINT SOLUTION. 2...BOTH REAL AND ADJOINT SOLUTION. |
| CD 4 | 19-24 | CHEBYSHEV ACCELERATION OF OUTER ITERATIONS. 0...YES, ACCELERATE THE OUTER ITERATIONS (DEFAULT). 1...NO ACCELERATION. |
| CD 5 | 25-30 | MINIMUM PLANE-BLOCK (RECORD) SIZE IN REAL*8 WORDS FOR I/O TRANSFER IN THE CONCURRENT INNER ITERATION STRATEGY. THE DEFAULT (=4500) IS HIGHLY RECOMMENDED. |
| CD 6 | 31-36 | OUTER ITERATION CONTROL. -3...BYPASS DIF3D MODULE. -2...CALCULATE DATA MANAGEMENT PARAMETERS AND PERFORM NEUTRONICS EDITS ONLY. -1...CALCULATE DATA MANAGEMENT PARAMETERS, CALCULATE OVERRELAXATION FACTORS AND PERFORM NEUTRONICS EDITS ONLY. GE.0...MAXIMUM NUMBER OF OUTER ITERATIONS (DEFAULT=30). |
| CD 7 | 37-42 | RESTART FLAG. 0...THIS IS NOT A RESTART (DEFAULT). 1...THIS IS A RESTART PROBLEM. |
| CD 8 | 43-48 | JOB TIME LIMIT, MAXIMUM (CP AND PP (OR WAIT)) PROCESSOR SECONDS (DEFAULT=1000000000). |
| CD 9 | 49-54 | NUMBER OF UPSCATTER ITERATIONS PER OUTER ITERATION (DEFAULT=5). PERTINENT TO UPSCATTER PROBLEMS ONLY. |
| CD 10 | 55-60 | CONCURRENT ITERATION EFFICIENCY OPTION. 0...PERFORM THE ESTIMATED NO. OF INNER ITERATIONS FOR EACH GROUP. 1...AVOID THE LAST PASS OF INNER ITERATIONS IN THOSE GROUPS FOR WHICH THE NO. OF ITERATIONS IN THE LAST PASS ARE LESS THAN A CODE DEPENDENT THRESHOLD. |

CD
 CD 11 61-66 ACCELERATION OF OPTIMUM OVERRELAXATION FACTOR
 CD CALCULATION.
 CD 0...NO ACCELERATION (DEFAULT).
 CD 1...ASYMPTOTIC SOURCE EXTRAPOLATION OF POWER ITERATIONS-
 CD USED TO ESTIMATE THE SPECTRAL RADIUS OF EACH INNER
 CD (WITHIN GROUP) ITERATION MATRIX.
 CD 12 67-72 OPTIMUM OVERRELAXATION FACTOR ESTIMATION ITERATION
 CD CONTROL. THE DEFAULT (=50) IS STRONGLY RECOMMENDED.
 C
 CN THE MAXIMUM NUMBER OF OUTER ITERATIONS SENTINEL
 CN SPECIFIES THE NUMBER OF OUTERS THAT CAN BE PERFORMED
 CN (COLS. 31-36) EACH TIME THE DIF3D MODULE IS INVOKED.
 CN
 CN THE DIF3D TERMINATION PROCEDURE WILL ALWAYS:
 CN 1...(RE)WRITE THE APPROPRIATE FLUX FILES
 CN (RTFLUX OR ATFLUX).
 CN 2...(RE)WRITE THE RESTART FILE DIF3D.
 CN TO FACILITATE AUTOMATIC RESTART, THE RESTART FLAG
 CN ON THE DIF3D RESTART CONTROL FILE WILL BE TURNED ON
 CN AUTOMATICALLY UPON DETECTION OF:
 CN 1...MAXIMUM NUMBER OF OUTER ITERATIONS.
 CN 2...TIME LIMIT.
 CN
 CN TO RESTART THE FLUX CALCULATION:
 CN EITHER
 CN
 CN PROVIDE THE RESTART DATA SET DIF3D AND
 CN THE APPROPRIATE FLUX DATA SET (RTFLUX OR ATFLUX)
 CN AND SPECIFY THEM UNDER "BLOCK=OLD" IN THE BCD
 CN INPUT DATA
 CN OR
 CN 1...SET THE RESTART FLAG (COLS. 37-42) TO 1 ON
 CN THE TYPE 03 CARD. THIS PERMITS IMMEDIATE
 CN RESUMPTION OF OUTER ITERATION ACCELERATION.
 CN 2...INCLUDE THE LATEST K-EFFECTIVE ESTIMATE
 CN (COLS. 13-24) AND THE DOMINANCE RATIO
 CN ESTIMATE ON THE TYPE 06 CARD (COLS. 61-72).
 CN 3...INCLUDE THE OPTIMUM OVERRELAXATION FACTORS
 CN FOR EACH GROUP (TYPE 07 CARD).
 CN 4...PROVIDE THE APPROPRIATE FLUX DATA SET (RTFLUX
 CN OR ATFLUX) AND SPECIFY IT UNDER "BLOCK=OLD"
 CN IN THE BCD INPUT DATA.
 CN
 CN A NON-ZERO TIME LIMIT (COLS. 43-48) OVERRIDES
 CN THE ACTUAL TIME LIMIT DETERMINED INTERNALLY
 CN BY SYSTEM ROUTINES IN THE ANL AND LBL PRODUCTION
 CN IMPLEMENTATIONS
 CN
 CN THE TIME LIMIT PARAMETER (COLS. 43-48) IS PERTINENT
 CN TO EACH ENTRY TO THE DIF3D MODULE.
 CN
 CN IT IS RECOMMENDED THAT AN ODD NUMBER OF UPSCATTER
 CN ITERATIONS BE SPECIFIED (COLS. 49-54) TO AVOID
 CN ADDITIONAL I/O OVERHEAD.
 CN
 CN THE USER IS CAUTIONED TO MONITOR THE POINT-WISE
 CN FISSION SOURCE CONVERGENCE TO ENSURE THAT MONOTONIC
 CN CONVERGENCE IS OBTAINED WHEN THE EFFICIENCY OPTION

CN (COLS. 55-60) IS ACTIVATED. -
 CN -
 CN THE OPTIMUM OVERRELAXATION FACTOR ACCELERATION OPTION -
 CN IS PRIMARILY INTENDED FOR PROBLEMS KNOWN TO HAVE HIGH -
 CN (>1.8) OPTIMUM OVERRELAXATION FACTORS. -
 CN -
 CN ITERATION CONTROL (COLS. 67-72) OF THE OPTIMUM -
 CN OVERRELAXATION FACTOR ESTIMATION IS PRIMARILY INTENDED -
 CN FOR USE IN CONJUNCTION WITH THE ASYMPTOTIC ACCELERATION-
 CN OPTION (COLS. 61-66). -
 C -
 C-----

C-----
 CR EDIT OPTIONS (TYPE 04) -
 C -
 CL FORMAT----- (I2,4X,10I6) -
 C -
 CD # COLUMNS CONTENTS...IMPLICATIONS, IF ANY -
 CD = =====
 CD 1 1-2 04 -
 CD -
 CD 2 7-12 PROBLEM DESCRIPTION EDIT (IN ADDITION TO USER INPUT -
 CD SPECIFICATIONS WHICH ARE ALWAYS EDITED. -
 CD 0...NO EDITS (DEFAULT). -
 CD 1...PRINT EDITS. -
 CD 2...WRITE EDITS TO AUXILIARY OUTPUT FILE. -
 CD 3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
 CD -
 CD 3 13-18 GEOMETRY (REGION TO MESH INTERVAL) MAP EDIT. -
 CD 0...NO EDITS (DEFAULT). -
 CD 1...PRINT EDITS. -
 CD 2...WRITE EDITS TO AUXILIARY OUTPUT FILE. -
 CD 3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
 CD -
 CD 4 19-24 GEOMETRY (ZONE TO MESH INTERVAL) MAP EDIT. -
 CD 0...NO EDITS (DEFAULT). -
 CD 1...PRINT EDITS. -
 CD 2...WRITE EDITS TO AUXILIARY OUTPUT FILE. -
 CD 3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
 CD -
 CD 5 25-30 MACROSCOPIC CROSS SECTION EDIT. -
 CD ENTER TWO DIGIT NUMBER SP WHERE -
 CD -
 CD S CONTROLS THE SCATTERING AND PRINCIPAL CROSS SECTIONS -
 CD P CONTROLS THE PRINCIPAL CROSS SECTIONS EDIT ONLY. -
 CD -
 CD THE INTEGERS S AND P SHOULD BE ASSIGNED ONE OF THE -
 CD FOLLOWING VALUES (LEADING ZEROES ARE IRRELEVANT). -
 CD 0...NO EDITS (DEFAULT). -
 CD 1...PRINT EDITS. -
 CD 2...WRITE EDITS TO AUXILIARY OUTPUT FILE. -
 CD 3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
 CD -
 CD 6 31-36 BALANCE EDITS -
 CD ENTER 3 DIGIT NUMBER GBR WHERE -
 CD -
 CD G CONTROLS GROUP BALANCE EDITS INTEGRATED OVER THE -
 CD REACTOR -
 CD

```

CD          B CONTROLS REGION BALANCE EDIT BY GROUP          -
CD          R CONTROLS REGION BALANCE EDIT TOTALS            -
CD          (INCLUDING NET PRODUCTION AND ENERGY MEDIANS)    -
CD                                                         -
CD          THE INTEGERS G, B, AND R SHOULD BE ASSIGNED ONE OF THE -
CD          FOLLOWING VALUES (LEADING ZEROES ARE IRRELEVANT) -
CD          0...NO EDITS (DEFAULT).                          -
CD          1...PRINT EDITS.                                  -
CD          2...WRITE EDITS TO AUXILIARY OUTPUT FILE.        -
CD          3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
CD                                                         -
CD 7      37-42  POWER EDITS                                  -
CD          ENTER 2 DIGIT NUMBER RM WHERE                    -
CD                                                         -
CD          R CONTROLS REGION POWER AND AVERAGE POWER DENSITY EDITS-
CD          M CONTROLS POWER DENSITY BY MESH INTERVAL EDIT (PWDINT)-
CD                                                         -
CD          THE INTEGERS R AND M SHOULD BE ASSIGNED          -
CD          ONE OF THE FOLLOWING VALUES (LEADING ZEROES ARE -
CD          IRRELEVANT)                                       -
CD          0...NO EDITS (DEFAULT).                          -
CD          1...PRINT EDITS.                                  -
CD          2...WRITE EDITS TO AUXILIARY OUTPUT FILE.        -
CD          3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
CD                                                         -
CD 8      43-48  FLUX EDITS                                  -
CD          ENTER 3 DIGIT INTEGER RMB WHERE                  -
CD                                                         -
CD          R CONTROLS FLUX EDIT BY REGION AND GROUP        -
CD          INCLUDING GROUP AND REGION TOTALS                -
CD          M CONTROLS TOTAL (GROUP INTEGRATED) FLUX EDIT   -
CD          BY MESH INTERVAL                                  -
CD          B CONTROLS TOTAL FLUX EDIT BY MESH INTERVAL AND GROUP -
CD          (RTFLUX OR ATFLUX)                               -
CD                                                         -
CD          THE INTEGERS R, M, AND B SHOULD BE ASSIGNED     -
CD          ONE OF THE FOLLOWING VALUES (LEADING ZEROES ARE -
CD          IRRELEVANT)                                       -
CD          0...NO EDITS (DEFAULT).                          -
CD          1...PRINT EDITS.                                  -
CD          2...WRITE EDITS TO AUXILIARY OUTPUT FILE.        -
CD          3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
CD                                                         -
CD 9      49-54  ZONE AVERAGED (REAL) FLUX EDIT.            -
CD          0...NO EDITS (DEFAULT).                          -
CD          1...PRINT EDITS.                                  -
CD          2...WRITE EDITS TO AUXILIARY OUTPUT FILE.        -
CD          3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
CD                                                         -
CD 10     55-60  REGION AVERAGED FLUX EDIT.                  -
CD          0...NO EDITS (DEFAULT).                          -
CD          1...PRINT EDITS.                                  -
CD          2...WRITE EDITS TO AUXILIARY OUTPUT FILE.        -
CD          3...WRITE EDITS TO BOTH PRINT AND AUXILIARY OUTPUT FILE-
CD                                                         -
CD 11     61-66  INTERFACE FILES TO BE WRITTEN IN ADDITION TO RTFLUX -
CD          AND/OR ATFLUX.                                    -
CD          ENTER 4 DIGIT INTEGER FSRP WHERE                -
CD                                                         -
CD          F CONTROLS WRITING OF SURFACE FAST FLUX TO SFEDIT -

```

CD S CONTROLS WRITING OF SURFACE POWER DENSITY TO SFEDIT -
 CD R CONTROLS WRITING OF RZFLUX -
 CD P CONTROLS WRITING OF PWDINT -
 CD -
 CD THE INTEGERS F, S, R, AND P SHOULD BE ASSIGNED ONE OF -
 CD THE FOLLOWING VALUES (LEADING ZEROES ARE IRRELEVANT) -
 CD 0...DO NOT WRITE THE INTERFACE FILE -
 CD 1...WRITE THE INTERFACE FILE (SFEDIT WILL BE WRITTEN -
 CD IN REGULAR MESH CELL ORDER) -
 CD 2...WRITE THE SFEDIT FILE IN REGION ORDER (PERTINENT -
 CD TO THE SFEDIT FILE ONLY) -
 CD -
 CD 12 67-72 MASTER DIF3D EDIT SENTINEL DURING CRITICALITY SEARCHES -
 CD -1...SUPPRESS ALL DIF3D EDITS EXCEPT THE ITERATION -
 CD HISTORY AND ERROR DIAGNOSTICS -
 CD 0...EDIT INPUT DATA ON 1ST SEARCH PASS, OUTPUT -
 CD INTEGRALS UPON CONVERGENCE OR UPON ACHIEVING THE -
 CD MAXIMUM SEARCH PASS LIMIT. -
 CD N...ALSO INVOKE SPECIFIED DIF3D EDITS EVERY N-TH -
 CD SEARCH PASS. -
 C -
 CN MULTI-DIGIT EDIT SPECIFICATION EXAMPLES. -
 CN -
 CN ENTERING THE INTEGER 201 IN COLUMNS 31-36 YIELDS -
 CN THE GROUP BALANCE EDIT ON THE AUXILIARY FILE AND -
 CN THE REGION BALANCE EDIT ON THE PRIMARY PRINT FILE. -
 CN -
 CN ENTERING THE INTEGER 30 IN COLUMNS 31-36 YIELDS -
 CN THE REGION BALANCE EDIT BY GROUP ON BOTH THE PRINT AND -
 CN THE AUXILIARY OUTPUT FILES. -
 CN -
 CN THE INTERFACE FILE SFEDIT CONTAINS SURFACE- AND -
 CN CELL-AVERAGED POWER DENSITY AND/OR FAST FLUX DATA -
 CN BY MESH CELL. ON OPTION IT IS WRITTEN IN EITHER -
 CN STANDARD FINE MESH CELL ORDER OR IN REGION ORDER.-
 C -
 C-----

C-----
 CR CONVERGENCE CRITERIA (TYPE 05) -
 C -
 CL FORMAT----- (I2,10X,3E12.5) -
 C -
 CD # COLUMNS CONTENTS...IMPLICATIONS, IF ANY -
 CD = =====
 CD 1 1-2 05 -
 CD -
 CD 2 13-24 EIGENVALUE CONVERGENCE CRITERION FOR STEADY STATE -
 CD CALCULATION (DEFAULT VALUE = 1.0E-7 IS RECOMMENDED). -
 CD -
 CD 3 25-36 POINTWISE FISSION SOURCE CONVERGENCE CRITERION -
 CD FOR STEADY STATE SHAPE CALCULATION -
 CD (DEFAULT VALUE = 1.0E-5 IS RECOMMENDED). -
 CD -
 CD 4 37-48 AVERAGE FISSION SOURCE CONVERGENCE CRITERION -
 CD FOR STEADY STATE SHAPE CALCULATION -
 CD (DEFAULT VALUE = 1.0E-5 IS RECOMMENDED). -
 C -
 CN IN UPSCATTERING PROBLEMS IT IS RECOMMENDED THAT -

CN THE EIGENVALUE CONVERGENCE CRITERION (COLS. 13-24) -
 CN BE .1 TIMES THE POINTWISE FISSION SOURCE CONVERGENCE -
 CN CRITERION (COLS. 25-36). -
 C -

C-----
 C-----
 CR OTHER FLOATING POINT DATA (TYPE 06) -
 C -

CL FORMAT----- (I2,10X,5E12.5) -
 C -

| CD # | COLUMNS | CONTENTS...IMPLICATIONS, IF ANY |
|------|---------|---|
| CD = | ===== | ===== |
| CD 1 | 1-2 | 06 |
| CD 2 | 13-24 | K-EFFECTIVE OF REACTOR (DEFAULT IS OBTAINED FROM THE APPROPRIATE RTFLUX OR ATFLUX FILE, IF PRESENT. OTHERWISE DEFAULT = 1.0). |
| CD 3 | 25-36 | ANY POINTWISE FISSION SOURCE WILL BE NEGLECTED IN THE POINTWISE FISSION SOURCE CONVERGENCE TEST IF IT IS LESS THAN THIS FACTOR TIMES THE R.M.S. FISSION SOURCE (DEFAULT VALUE = .001 IS RECOMMENDED). |
| CD 4 | 37-48 | ERROR REDUCTION FACTOR TO BE ACHIEVED BY EACH SERIES OF INNER ITERATIONS FOR EACH GROUP DURING A SHAPE CALCULATION - STRONGLY RECOMMENDED THAT THE DEFAULT VALUE OF (.04) BE USED. |
| CD 5 | 49-60 | STEADY STATE REACTOR POWER (WATTS). (DEFAULT = 1.0). |
| CD 6 | 61-72 | DOMINANCE RATIO (FOR RESTART JOBS ONLY). |

C
 CN K-EFFECTIVE SPECIFICATIONS (COLS. 13-24): -
 CN 1...FOR K-EFFECTIVE PROBLEMS, SUPPLY ESTIMATED -
 CN K-EFFECTIVE OF REACTOR. -
 CN 2...FOR RESTARTED K-EFFECTIVE PROBLEMS, SUPPLY -
 CN LATEST K-EFFECTIVE ESTIMATE SUPPLIED ON THE -
 CN ITERATION HISTORY EDIT. -
 CN 3...FOR SOURCE PROBLEMS, SUPPLY K-EFFECTIVE OF -
 CN THE REACTOR. -
 CN DEFAULT IS OBTAINED FROM THE APPROPRIATE RTFLUX OR -
 CN ATFLUX FILE, IF PRESENT. OTHERWISE DEFAULT=1.0 . -
 C -

CN NON-MONOTONIC POINTWISE FISSION SOURCE CONVERGENCE -
 CN IS USUALLY INDICATIVE OF THE NEED TO TIGHTEN THE ERROR -
 CN REDUCTION FACTOR(COLS. 37-48). THIS IS FREQUENTLY TRUE -
 CN IN TRIANGULAR GEOMETRY PROBLEMS WHERE A VALUE OF .01 IS -
 CN USUALLY SUFFICIENT TO OBTAIN MONOTONIC CONVERGENCE. -
 C-----

C-----
 C-----
 CR OPTIMUM OVERRELAXATION FACTORS (TYPE 07) -
 C -

CL FORMAT----- (I2,10X,5E12.5) -
 C -

| CD # | COLUMNS | CONTENTS...IMPLICATIONS, IF ANY |
|------|---------|---------------------------------|
| CD = | ===== | ===== |

CD 1 1-2 07
 CD
 CD 2 13-24 OPTIMUM OVERRELAXATION FACTOR FOR GROUP 1.
 CD
 CD 3 25-36 OPTIMUM OVERRELAXATION FACTOR FOR GROUP 2.
 CD
 CD 4 37-48 OPTIMUM OVERRELAXATION FACTOR FOR GROUP 3.
 CD
 CD 5 49-60 OPTIMUM OVERRELAXATION FACTOR FOR GROUP 4.
 CD
 CD 6 61-72 OPTIMUM OVERRELAXATION FACTOR FOR GROUP 5.
 C
 CN REPEAT 5 VALUES PER CARD FOR AS MANY TYPE 07 CARDS
 CN AS ARE NEEDED.
 CN
 CN THE OPTIMUM OVERRELAXATION FACTORS ARE NORMALLY
 CN OBTAINED FROM THE RESTART INSTRUCTIONS PRINTED
 CN IMMEDIATELY AFTER THE DIF3D ITERATION HISTORY EDIT.
 CN IN THE RESTART INSTRUCTIONS, THE FACTORS ARE ALWAYS
 CN EDITED IN THE --REAL PROBLEM-- ORDERING AND SHOULD BE
 CN ENTERED ON THE TYPE 07 CARD --EXACTLY-- AS EDITED
 CN IN THE RESTART INSTRUCTIONS.
 CN
 CN THE PERMISSIBLE FACTOR RANGE IS BOUNDED BY 1.0 AND 2.0
 CN INCLUSIVE. A ZERO OR BLANK FACTOR ENTRY DEFAULTS
 CN TO 1.0. FACTORS ARE COMPUTED FOR THOSE GROUPS HAVING
 CN A FACTOR OF 1.0; FACTORS GREATER THAN 1.0 ARE NOT
 CN RECOMPUTED.
 CN
 CN TYPE 07 CARDS ARE PRIMARILY INTENDED FOR RESTART JOBS
 CN ONLY (STRONGLY RECOMMENDED).
 C
 C-----

C-----
 CR NEAR CRITICAL SOURCE PROBLEM ASYMPTOTIC EXTRAPOLATION
 CR PARAMETERS (TYPE 08)
 C
 CC ***** WARNING...SELECT THIS OPTION ONLY IF THE *****
 CC ***** ASYMPTOTIC EXTRAPOLATION IS REQUIRED FOR *****
 CC ***** THIS PROBLEM. *****
 C
 CL FORMAT----- (I2,4X,I6,E12.5,I6)
 C

| CD # | COLUMNS | CONTENTS...IMPLICATIONS, IF ANY |
|------|---------|---|
| CD = | ===== | ===== |
| CD 1 | 1-2 | 08 |
| CD 2 | 7-12 | NUMBER OF OUTER (POWER) ITERATIONS PERFORMED PRIOR TO ASYMPTOTIC EXTRAPOLATION OF NEAR CRITICAL SOURCE PROBLEM (DEFAULT=5). |
| CD 3 | 13-24 | EIGENVALUE OF THE HOMOGENEOUS PROBLEM CORRESPONDING TO THE NEAR CRITICAL SOURCE PROBLEM. THIS EIGENVALUE MUST BE LESS THAN ONE. |
| CD 4 | 25-30 | INITIAL FLUX GUESS SENTINEL. 0...FLAT FLUX GUESS=1.0 (DEFAULT) 1...FLAT FLUX GUESS=0.0 |

C
 CN THE TYPE 08 CARD IS REQUIRED TO ACTIVATE AN ALTERNATE
 CN SPECIAL ACCELERATION SCHEME FOR NEAR CRITICAL
 CN SOURCE PROBLEMS.
 CN
 CN IF COLS. 13-24 ARE ZERO OR BLANK, THE HOMOGENEOUS
 CN PROBLEM EIGENVALUE WILL BE ESTIMATED. IN THIS CASE, IT
 CN IS RECOMMENDED TO INCREASE THE NUMBER OF ITERATIONS IN
 CN COLS. 7-12 TO AT LEAST 10.
 C
 C-----

C-----
 CR SN TRANSPORT OPTIONS (TYPE 09)
 C
 CL FORMAT----- (I2,4X,2I6,6X,E12.4)
 C
 CD # COLUMNS CONTENTS...IMPLICATIONS, IF ANY
 CD = =====
 CD 1 1-2 09
 CD
 CD 2 7-12 SN ORDER.
 CD
 CD 3 13-18 MAXIMUM ALLOWED NUMBER OF LINE SWEEPS PER LINE PER
 CD INNER ITERATION (DEFAULT=10).
 CD
 CD 4 25-36 LINE SWEEP CONVERGENCE CRITERION (DEFAULT=1.0E-4).
 C
 CN TO INVOKE THE DIF3D TRANSPORT OPTION, THE TYPE 09 CARD
 CN MUST BE PRESENT WITH A NONZERO SN ORDER. FOR THE TIME
 CN BEING, USERS MUST ALSO CONTINUE TO 'PRELIB' TO
 CN DATASET 'C116.B99983.MODLIB' TO INVOKE THIS OPTION.
 C
 C-----

C-----
 CR PARAMETERS FOR NODAL OPTION (TYPE 10)
 C
 CL FORMAT----- (I2,4X,7I6)
 C
 CD # COLUMNS CONTENTS...IMPLICATIONS, IF ANY
 CD = =====
 CD 1 1-2 10
 CD
 CD 2 7-12 NODAL APPROXIMATION IN XY-PLANE.
 CD ENTER 3 DIGIT NUMBER LMN WHERE
 CD
 CD L DETERMINES WHETHER THIS IS A DIFFUSION OR TRANSPORT
 CD CALCULATION.
 CD M IS THE ORDER OF THE POLYNOMIAL APPROXIMATION TO THE
 CD ONE-DIMENSIONAL FLUXES IN THE XY-PLANE.
 CD N IS THE ORDER OF THE POLYNOMIAL APPROXIMATION TO THE
 CD LEAKAGES TRANSVERSE TO THE X- AND Y-DIRECTIONS.
 CD
 CD HEXAGONAL GEOMETRY:
 CD L = 0... (ALWAYS - ONLY DIFFUSION THEORY IS AVAILABLE
 CD IN HEXAGONAL GEOMETRY).
 CD M = 2...NH2 FLUX APPROXIMATION.
 CD


```

CD          M = 3...NH3 FLUX APPROXIMATION.
CD          M = 4...NH4 FLUX APPROXIMATION (DEFAULT).
CD          N = 0...(ALWAYS).
CD
CD          CARTESIAN GEOMETRY:
CD          L = 0...DIFFUSION-THEORY OPTION (DEFAULT).
CD          L = 1...TRANSPORT-THEORY OPTION.
CD          M = 2...NX2 (QUADRATIC) FLUX APPROXIMATION.
CD          M = 3...NX3 (CUBIC ) FLUX APPROXIMATION (DEFAULT).
CD          M = 4...NX4 (QUARTIC ) FLUX APPROXIMATION.
CD          N = 0...CONSTANT LEAKAGE APPROXIMATION.
CD          N = 2...QUADRATIC LEAKAGE APPROXIMATION (DEFAULT).
CD
CD          LEADING ZEROS ARE IRRELEVANT.
CD          THEREFORE, DEFAULT VALUES FOR MN ARE 40 (HEXAGONAL
CD          GEOMETRY) AND 32 (CARTESIAN GEOMETRY).
CD
CD          IF THE TRANSPORT OPTION (L=1) IS SPECIFIED, TRANSPORT
CD          THEORY IS USED IN BOTH THE XY-PLANE AND THE AXIAL
CD          DIRECTION IN THREE-DIMENSIONAL CARTESIAN GEOMETRY.
CD
CD 3 13-18  NODAL APPROXIMATION IN Z-DIRECTION.
CD          ENTER 2 DIGIT NUMBER MN WHERE
CD
CD          M IS THE ORDER OF THE POLYNOMIAL APPROXIMATION TO THE
CD          ONE-DIMENSIONAL FLUX IN THE Z-DIRECTION.
CD          N IS THE ORDER OF THE POLYNOMIAL APPROXIMATION TO THE
CD          LEAKAGE TRANSVERSE TO THE Z-DIRECTION.
CD
CD          HEXAGONAL AND CARTESIAN GEOMETRIES:
CD          M = 2...NZ2 (QUADRATIC) FLUX APPROXIMATION.
CD          M = 3...NZ3 (CUBIC ) FLUX APPROXIMATION (DEFAULT).
CD          M = 4...NZ4 (QUARTIC ) FLUX APPROXIMATION (CARTESIAN
CD          GEOMETRY ONLY).
CD          N = 0...CONSTANT LEAKAGE APPROXIMATION.
CD          N = 2...QUADRATIC LEAKAGE APPROXIMATION (DEFAULT).
CD
CD          LEADING ZEROS ARE IRRELEVANT.
CD          THEREFORE, DEFAULT VALUE FOR MN IS 32.
CD
CD 4 19-24  COARSE-MESH REBALANCE ACCELERATION CONTROL.
CD          -1...NO COARSE-MESH REBALANCE ACCELERATION.
CD          .GT.0...NUMBER OF FINE MESH PER REBALANCE MESH IN X- AND
CD          Y-DIRECTIONS - CARTESIAN GEOMETRY ONLY (DEFAULT=4).
CD
CD 5 25-30  NUMBER OF XY-PLANE PARTIAL CURRENT SWEEPS PER GROUP
CD          PER AXIAL MESH SWEEP PER OUTER ITERATION.
CD          (DEFAULT = 0 - LET CODE DECIDE).
CD
CD 6 31-36  NUMBER OF AXIAL PARTIAL CURRENT SWEEPS PER GROUP
CD          PER AXIAL PARTIAL CURRENT SWEEP
CD          PER OUTER ITERATION (DEFAULT=2).
CD
CD 7 37-42  HALF-DOMAIN SYMMETRY FLAG.
CD          -1...DO NOT USE 30 DEGREE (HEXAGONAL GEOMETRY) OR 45
CD          DEGREE (CARTESIAN GEOMETRY) PLANAR SYMMETRY EVEN
CD          IF SUCH SYMMETRY EXISTS.
CD          0...USE 30 DEGREE (HEXAGONAL GEOMETRY) OR 45 DEGREE
CD          (CARTESIAN GEOMETRY) PLANAR SYMMETRY IF SUCH
CD          SYMMETRY EXISTS (DEFAULT).

```

C
 CN THE NODAL OPTION IS INVOKED IN HEXAGONAL GEOMETRY BY
 CN SPECIFYING GEOMETRY-TYPE SENTINELS BETWEEN 110 AND 128
 CN ON THE A.NIP3 TYPE 03 CARD.
 CN
 CD 8 43-48 ASYMPTOTIC SOURCE EXTRAPOLATION SENTINEL.
 CD 0...PERFORM ASYMPTOTIC SOURCE EXTRAPOLATION ON THE
 CD THE NODAL OUTER ITERATIONS.
 CD 1...DO NOT PERFORM ASYMPTOTIC SOURCE EXTRAPOLATION
 CD
 CN THE NODAL OPTION IS INVOKED IN CARTESIAN GEOMETRY BY
 CN SPECIFYING GEOMETRY-TYPE SENTINELS 40 OR 44 ON THE
 CN A.NIP3 TYPE 03 CARD AND PROVIDING ANY ACCEPTABLE
 CN (E.G. DEFAULT) VALUES ON A.DIF3D TYPE 10 CARD.
 CN
 CN *** THE CARTESIAN-GEOMETRY NODAL OPTION MAY NOT BE
 CN AVAILABLE IN ALL VERSIONS OF DIF3D. ***
 C
 CN IT IS IMPORTANT THAT THE NUMBER OF FINE MESH PER
 CN REBALANCE MESH BE CHOSEN SUCH THAT THE AVERAGE
 CN REBALANCE MESH SPACING IS APPROXIMATELY 30 TO 40 CM IN
 CN THE XY-PLANE. THUS, FOR EXAMPLE, IF THE AVERAGE FINE
 CN MESH SPACING IS DELTA CM, THEN THE INTEGER INPUT IN
 CN COLS. 19-24 SHOULD BE BETWEEN 30/DELTA AND 40/DELTA.
 C
 CN IF SLOW (OR DIVERGENT) ITERATIVE CONVERGENCE BEHAVIOR
 CN IS OBSERVED, THE NUMBER OF PARTIAL CURRENT SWEEPS
 CN SPECIFIED IN COLS. 25-30 AND 31-36 SHOULD BE INCREASED.
 C
 C-----

C-----
 CR AXIAL COARSE-MESH REBALANCE BOUNDARIES FOR NODAL
 CR OPTION (TYPE 11)
 C
 CL FORMAT----- (I2,10X,3(I6,E12.5))
 C

| CD # | COLUMNS | CONTENTS...IMPLICATIONS, IF ANY |
|------|---------|---|
| CD = | ===== | ===== |
| CD 1 | 1-2 | 11 |
| CD 2 | 13-18 | NUMBER OF AXIAL COARSE-MESH REBALANCE INTERVALS. |
| CD 3 | 19-30 | UPPER Z-COORDINATE OF THE COARSE-MESH REBALANCE BOUNDARY. |
| CD 4 | 31-36 | NUMBER OF AXIAL COARSE-MESH REBALANCE INTERVALS. |
| CD 5 | 37-42 | UPPER Z-COORDINATE OF THE COARSE-MESH REBALANCE BOUNDARY. |
| CD 6 | 49-54 | NUMBER OF AXIAL COARSE-MESH REBALANCE INTERVALS. |
| CD 7 | 55-66 | UPPER Z-COORDINATE OF THE COARSE-MESH REBALANCE BOUNDARY. |

C
 CN THE TYPE 11 CARD IS PERTINENT ONLY WHEN THE THREE-
 CN DIMENSIONAL NODAL OPTION (A.NIP3 TYPE 03 GEOMETRY-TYPE
 CN SENTINEL VALUE EQUAL TO 44 OR BETWEEN 120 AND 128) IS
 CN

CN SPECIFIED. -
 CN -
 CN IF NO TYPE 11 CARDS ARE PRESENT, THE AXIAL COARSE-MESH -
 CN REBALANCE BOUNDARIES ARE DEFINED BY THE AXIAL COARSE- -
 CN MESH BOUNDARIES OBTAINED FROM THE GEODST FILE. THESE -
 CN BOUNDARIES IN TURN ARE ANY BOUNDARY POSITIONS SPECIFIED-
 CN ON THE DATASET A.NIP3 TYPE 09 OR 30 CARDS. -
 CN -
 CN AXIAL COARSE-MESH REBALANCE BOUNDARIES MUST BE SELECTED-
 CN FROM THE SET OF COARSE-MESH BOUNDARIES CONTAINED IN THE-
 CN GEODST FILE, AS DETERMINED BY THE COARSE-MESH -
 CN BOUNDARIES WHICH ARE EXPLICITLY MENTIONED ON THE -
 CN DATASET A.NIP TYPE 09 OR 30 CARDS. -
 CN -
 CN BOUNDARIES ARE SPECIFIED VIA NUMBER PAIRS. -
 CN EACH NUMBER PAIR IS OF THE FORM (N(I), Z(I)). THERE -
 CN ARE N(I) AXIAL COARSE-MESH REBALANCE INTERVALS BETWEEN -
 CN Z(I-1) AND Z(I), WHERE Z(0) IS THE LOWER REACTOR -
 CN BOUNDARY IN THE Z-DIRECTION. NUMBER PAIRS MUST BE -
 CN GIVEN IN ORDER OF INCREASING MESH COORDINATES. ALL -
 CN AXIAL COARSE-MESH REBALANCE BOUNDARIES MUST COINCIDE -
 CN WITH THE MESH LINES WHICH BOUND MESH INTERVALS. -
 C -
 C-----

C-----
 CR PARAMETERS FOR VARIATIONAL NODAL OPTION (TYPE 10) -
 C -
 CL FORMAT----- (I2,4X,11I6) -
 C -
 CD # COLUMNS CONTENTS...IMPLICATIONS, IF ANY -
 CD = ===== -
 CD 1 1-2 12 -
 CD -
 CD 2 7-12 NODAL SPATIAL APPROXIMATION. -
 CD ENTER 3 DIGIT NUMBER LMN WHERE -
 CD -
 CD L IS THE ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
 CD SOURCE WITHIN THE NODE. -
 CD M IS THE ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
 CD FLUXES WITHIN THE NODE. -
 CD N IS THE ORDER OF THE POLYNOMIAL APPROXIMATION OF THE -
 CD LEAKAGES ON THE SURFACES OF THE NODES. -
 CD -
 CD HEXAGONAL AND CARTESIAN GEOMETRY: -
 CD L = 1... LINEAR SOURCE APPROXIMATION. -
 CD L = 2... QUADRATIC SOURCE APPROXIMATION. -
 CD L = 3... CUBIC SOURCE APPROXIMATION. -
 CD L = 4... QUARTIC SOURCE APPROXIMATION. -
 CD L = 5... 5TH ORDER SOURCE APPROXIMATION. -
 CD L = 6... 6TH ORDER SOURCE APPROXIMATION. -
 CD (DEFAULT VALUE L=N+1). -
 CD (L CANNOT BE GREATER THAN M). -
 CD M = 1... LINEAR FLUX APPROXIMATION. -
 CD M = 2... QUADRATIC FLUX APPROXIMATION. -
 CD M = 3... CUBIC FLUX APPROXIMATION. -
 CD M = 4... QUARTIC FLUX APPROXIMATION (DEFAULT) -
 CD M = 5... 5TH ORDER FLUX APPROXIMATION. -
 CD M = 6... 6TH ORDER FLUX APPROXIMATION. -
 CD -

```

CD          N = 0... FLAT LEAKAGE APPROXIMATION.
CD          N = 1... LINEAR LEAKAGE APPROXIMATION (DEFAULT).
CD          N = 2... QUADRATIC LEAKAGE APPROXIMATION.
CD
CD
CD          LEADING ZEROS ARE IRRELEVANT.
CD          THEREFORE, DEFAULT VALUES FOR LMN ARE 241
CD          M = 5 OR 6 ONLY FOR HEXAGONAL GEOMETRY.
CD          IN 3D HEXAGONAL GEOMETRY M = 5 OR 6 PROVIDES
CD          FULL EXPANSION IN THE X AND Y PLANE, Z DIRECTION
CD          IS EXPANDED TO 4TH ORDER.
CD
CD
CD 3 13-18  ANGULAR APPROXIMATION.
CD          ENTER 2 DIGIT NUMBER MN WHERE
CD
CD          M IS THE ORDER OF THE PN EXPANSION OF THE FLUX.
CD          N IS THE ORDER OF THE PN EXPANSION OF THE LEAKAGE.
CD
CD          HEXAGONAL AND CARTESIAN GEOMETRIES:
CD          M = 1... P1 FLUX EXPANSION.
CD          M = 3... P3 FLUX EXPANSION (DEFAULT).
CD          M = 5... P5 FLUX EXPANSION
CD          N = 1... P1 LEAKAGE EXPANSION.
CD          N = 3... P3 LEAKAGE EXPANSION (DEFAULT).
CD          N = 5... P5 LEAKAGE EXPANSION
CD
CD          LEADING ZEROS ARE IRRELEVANT.
CD          THEREFORE, DEFAULT VALUE FOR MN IS 33.
CD          MN EQUAL TO 11 CORRESPONDS TO DIFFUSION CALCULATION.
CD          IF MN IS NEGATIVE, SIMPLIFIED SPHERICAL HARMONICS
CD          ARE USED.
CD
CD 4 19-24  COARSE-MESH REBALANCE ACCELERATION CONTROL.
CD          -1...NO COARSE-MESH REBALANCE ACCELERATION.
CD          .GT.0...NUMBER OF FINE MESH PER REBALANCE MESH IN X- AND
CD          Y-DIRECTIONS - CARTESIAN GEOMETRY ONLY (DEFAULT=6).
CD
CD 5 25-30  NUMBER OF XY-PLANE PARTIAL CURRENT SWEEPS PER GROUP
CD          PER AXIAL MESH SWEEP PER OUTER ITERATION.
CD          (DEFAULT = 0 - LET CODE DECIDE).
CD
CD 6 31-36  NUMBER OF AXIAL PARTIAL CURRENT SWEEPS PER GROUP
CD          PER AXIAL PARTIAL CURRENT SWEEP
CD          PER OUTER ITERATION (DEFAULT=0 - LET CODE DECIDE)
CD
CD 7 37-42  HALF-DOMAIN SYMMETRY FLAG.
CD          -1...DO NOT USE 30 DEGREE (HEXAGONAL GEOMETRY) OR 45
CD          DEGREE (CARTESIAN GEOMETRY) PLANAR SYMMETRY EVEN
CD          IF SUCH SYMMETRY EXISTS.
CD          0...USE 30 DEGREE (HEXAGONAL GEOMETRY) OR 45 DEGREE
CD          (CARTESIAN GEOMETRY) PLANAR SYMMETRY IF SUCH
CD          SYMMETRY EXISTS (DEFAULT).
C
CN          THE NODAL OPTION IS INVOKED IN HEXAGONAL GEOMETRY BY
CN          SPECIFYING GEOMETRY-TYPE SENTINELS BETWEEN 110 AND 128
CN          ON THE A.NIP3 TYPE 03 CARD.
CD 8 43-48  ASYMPTOTIC SOURCE EXTRAPOLATION SENTINEL.
CD          -1...PERFORM ASYMPTOTIC SOURCE EXTRAPOLATION ON THE

```

```

CD          NODAL OUTER ITERATIONS ONLY ON FISSION SOURCES. -
CD          NO EXTRA-SPACE IS NEEDED TO STORE PREVIOUS OUTER -
CD          ITERATION CURRENTS. -
CD          0...PERFORM ASYMPTOTIC SOURCE EXTRAPOLATION ON THE -
CD          NODAL OUTER ITERATIONS ON FISSION SOURCES AND -
CD          CURRENTS. -
CD          1...DO NOT PERFORM ASYMPTOTIC SOURCE EXTRAPOLATION -
CD          -
CD 9    49-54  ANISOTROPIC SCATTERING APPROXIMATION NPNO. -
CD          0...ISOTROPIC SCATTERING (DEFAULT). -
CD          N...ANISOTROPIC SCATTERING ORDER (.LE.3). -
CD          N MUST BE LESS THAN OR EQUAL TO MAXORD, MAXIMUM -
CD          ANISOTROPIC ORDER SPECIFIED IN ISOTXS OR COMPKS -
CD          FILES. -
CD          -
CD 10   55-60  EXTENDED TRANSPORT APPROXIMATION (NXTR) ON TOTAL -
CD          CROSS SECTION. -
CD          -1...IF NPNO .EQ. 0 USE TOTAL CROSS SECTION PROVIDED -
CD          IN COMPKS FILE, OTHERWISE USE TRANSPORT CROSS -
CD          SECTION INSTEAD OF TOTAL ONE. -
CD          0...(DEFAULT). -
CD          IF NPNO .EQ. 0 USE TRANSPORT CROSS SECTION -
CD          PROVIDED IN COMPKS FILE. -
CD          IF NPNO .GT. 0 AND NPNO .EQ. MAXORD USE TOTAL -
CD          CROSS SECTION PROVIDED IN COMPKS FILE. -
CD          IF NPNO .GT. 0 AND NPNO .LT. MAXORD CORRECT TOTAL -
CD          CROSS SECTION PROVIDED IN COMPKS FILE WITH -
CD          EXTENDED TRANSPORT APPROXIMATION TAKING INTO -
CD          ACCOUNT THE NPNO + 1 ORDER SCATTERING CROSS -
CD          SECTIONS (BHS APPROXIMATION). -
CD          N...IF NXTR .LE. NPNO USE TOTAL CROSS SECTION. -
CD          IF NXTR .GT. NPNO PERFORM EXTENDED TRANSPORT -
CD          APPROXIMATION ON TOTAL CROSS SECTION FROM NPNO + 1 -
CD          TO NXTR ORDER. -
CD          -
CD 11   61-66  NODAL COUPLING COEFFICIENT PACKING OPTION. -
CD          0...NO PACKING WILL BE PERFORMED UNLESS NOT ENOUGH -
CD          ECM MEMORY IS AVAILABLE (DEFAULT). -
CD          1...NODAL COUPLING COEFFICIENT WILL BE PACKED (ONLY -
CD          UNIQUE ELEMENTS ARE STORED). THIS OPTION SHOULD -
CD          BE USED, ESPECIALLY ON WORKSTATIONS, WHEN IT WILL -
CD          ALLOW THE PROBLEM TO RUN WITH ALL GROUP FLUXES -
CD          AND CURRENTS IN CORE. -
CD          -
CD 12   67-72  RADIAL INNER ITERATION ALGORITHM. -
CD          0...PARTITIONED MATRIX ALGORITHM (DEFAULT). -
CD          1...FULL MATRIX ALGORITHM. THIS ALGORITHM IS SOMETIMES -
CD          NECESSARY WITH VERY SMALL NODE MESH SIZE WHERE -
CD          DIVERGENCE CAN OCCUR. THIS ALGORITHM REQUIRES A -
CD          SIGNIFICANTLY LARGER COMPUTATIONAL TIME. -
CD          FULL MATRIX ALGORITHM IS IMPOSED WHEN ONLY ONE -
CD          OUTER ITERATION IS SPECIFIED (FIXED SOURCE -
CD          PROBLEM). -
CD          -
CN          THE NODAL OPTION IS INVOKED IN CARTESIAN GEOMETRY BY -
CN          SPECIFYING GEOMETRY-TYPE SENTINELS 40 OR 44 ON THE -
CN          A.NIP3 TYPE 03 CARD AND PROVIDING ANY ACCEPTABLE -
CN          (E.G. DEFAULT) VALUES ON A.DIF3D TYPE 12 CARD. -
CN          -
CN          *** THE CARTESIAN-GEOMETRY NODAL OPTION MAY NOT BE -

```

CN AVAILABLE IN ALL VERSIONS OF DIF3D. *** -
C -
CN IT IS IMPORTANT THAT THE NUMBER OF FINE MESH PER -
CN REBALANCE MESH BE CHOSEN SUCH THAT THE AVERAGE -
CN REBALANCE MESH SPACING IS APPROXIMATELY 30 TO 40 CM IN -
CN THE XY-PLANE. THUS, FOR EXAMPLE, IF THE AVERAGE FINE -
CN MESH SPACING IS DELTA CM, THEN THE INTEGER INPUT IN -
CN COLS. 19-24 SHOULD BE BETWEEN 30/DELTA AND 40/DELTA. -
C -
CN IF SLOW (OR DIVERGENT) ITERATIVE CONVERGENCE BEHAVIOR -
CN IS OBSERVED, THE NUMBER OF PARTIAL CURRENT SWEEPS -
CN SPECIFIED IN COLS. 25-30 AND 31-36 SHOULD BE INCREASED.-
C -
C-----

CEOF

Distribution for ANL-95/40

Internal:

| | | |
|---------------------|------------------|-------------------|
| C. H. Adams | K. N. Grimm | A. P. Olson |
| S. K. Bhattacharyya | M. R. Hale | Y. Orechwa |
| R. N. Blomquist | N. A. Hanan | G. Palmiotti (36) |
| M. M. Bretscher | U. R. Hanebutte | R. B. Pond |
| L. L. Briggs | D. J. Hill | E. A. Rhodes |
| K. A. Bunde | R. N. Hill | R. W. Schaefer |
| J. E. Cahalan | R. N. Hwang | D. M. Smith |
| Y. I. Chang | G. R. Imel | C. G. Stenberg |
| J. M. Carpenter | K. Laurin-Kovitz | J. A. Stillman |
| B. R. Chandler | H. S. Khalil | C. E. Till |
| R. J. Cornella | R. M. Lell | B. J. Toppel |
| J. R. Deen | J. J. Liaw | R. B. Turski |
| K. L. Derstine | M. J. Lineberry | D. C. Wade |
| T. Fanning | D. M. Malon | D. P. Weber |
| E. K. Fujita | J. E. Matos | R. A. Wigeland |
| E. M. Gelbard | S. C. Mo | W. L. Woodruff |
| R. W. Goin | R. D. McKnight | A. M. Yacout |
| G. L. Grasseschi | | |

External:

DOE-OSTI, for distribution per UC-534 (60)
Manager, Chicago Operations Office
ANL-W Library
ANL-E Library

Reactor Analysis and Safety Division Review Committee:

R. O. Anderson, Northern States Power Company, Minneapolis
M. L. Corradini, University of Wisconsin, Madison
A. F. Henry, Massachusetts Institute of Technology, Cambridge
J. C. Lee, University of Michigan, Ann Arbor
V. H. Ransom, Purdue University, West Lafayette

D. T. Ingersoll, Leader, Nuclear Analysis & Shielding Sect., Oak Ridge Natl. Lab., Oak Ridge, TN
Division Leader, N-Division, Los Alamos National Laboratory, NM
A. E. Walter, Safety and Control Technology, Westinghouse Hanford Co., Richland, WA (2)
Information Manager, Nuclear Safety Library, Brookhaven Natl. Lab., Upton, NY (2)
Manager, Reactor Safety Research Department, Sandia Natl. Laboratories, Albuquerque, NM (2)
R. S. Denning, Nuclear Facility Safety, Battelle Memorial Institute, Columbus, OH
Manager, Safety Engineering, Westinghouse Electric Corporation ARD, Madison, PA (2)
Manager, LMFBR Physics and Safety, Rockwell International, Downey, CA (2)
P. M. Magee, Manager, Design Analysis, General Electric, San Jose, CA (2)
Electric Power Research Institute, LMFBR Group, Palo Alto, CA
Chief, Severe Accident Issues Branch, NRC/RES/SAIB, Rockville, MD (2)
Chief, Accident Evaluation Branch, NRC/RES/DSR, Rockville, MD
Executive Secretary, NRC-ACRS, Bethesda, MD (5)
Information Services, Babcock & Wilcox Co., Lynchburg, VA
X6 Group Leader, Mail Stop B226, Los Alamos Natl. Lab., Los Alamos, NM

External: (Cont'd.)

Manager, ARP Nuclear Engineering, Knolls Atomic Power Laboratory, Schenectady, NY
M. Natelson, Manager, Reactor Technology, Bettis Atomic Power Laboratory, West Mifflin, PA
B. Worley, Reactor Analysis, Engineering Physics and Mathematics Div., ORNL, Oak Ridge, TN
C. Cowan, General Electric, San Jose, CA
J. W. Daughtry, Westinghouse Hanford Co., Richland, WA
P. W. Dickson, Westinghouse Savannah River Laboratory, Aiken, SC
R. Doncals, Westinghouse Electric Corp., Madison, PA
J. J. Dorning, University of Virginia, Charlottesville, VA
L. D. Felten, Rockwell International Corp., Canoga Park, CA
N. C. Francis, Knolls Atomic Power Laboratory, Schenectady, NY
J. Kallfelz, PSI, Switzerland
T. S. Kress, Oak Ridge National Laboratory, Oak Ridge, TN
J. Lake, EG&G Idaho, Inc., Idaho Falls, ID
E. Lewis, Northwestern University, Evanston, IL (5)
M. R. Mendelson, Knolls Atomic Power Laboratory, Schenectady, NY
W. F. Miller, Jr., Los Alamos National Laboratory, Los Alamos, NM
K. Ott, Purdue University, West Lafayette, IN
S. Pearlstein, Brookhaven National Laboratory, Upton, NY
Tom Downar, Dept. of Nuclear Engineering, Purdue University, West Lafayette, IN
Joel Rhodes, Studsvik, of America, Idaho Falls, ID
C. Apperson, Reactor Physics Group, Westinghouse Savannah River Co., Aiken, SC
Wesley Davis, Babcock & Wilcox, Space and Defense Systems, Lynchburg, VA
Salim Jahsham, EG&G Idaho, Inc./INEL, Idaho Falls, ID
Wm. Martin, Dept. of Nucl. Eng., Univ. of Michigan, Cooley Bldg., No. Campus, Ann Arbor, MI
James R. Thomas, Mechanical Engineering Dept., VPI&SU, Blacksburg, VA
Steven Rowe, Illinois Power Company, Clinton Power Station, Clinton, IL
Del Pallotta, Commonwealth Edison Co., Downers Grove, IL
Jasmina Vujic, University of California, Dept. of Nucl. Eng., Berkeley, CA
A. J. H. Goddard, Imperial College, University of London, UK
T. Takeda, Osaka University, Osaka, Japan
E. H. Mund, Universite' Libre De Bruxelles, Belgium
Edward Larsen, University of Michigan, Ann Arbor, MI
Paul Nelson, Texas A&M University, College Station, TX
Marvin Adams, Texas A&M University, College Station, TX
Alireza Haghghat, Pennsylvania State University, University Park, PA
Paul Turinsky, North Carolina State University, Raleigh, NC
Y. Ronen, Ben Gurion University, Beer-Sheva, Israel
T. A. Germogenova, Keldysh Institute, Moscow, Russia
H. L. Dodds, University of Tennessee, Knoxville, TN
D. G. Cacuci, KfK, Karlsruhe, Germany
H. Wider, European Research Center, Ispra, Italy
H. Finnemann, Siemens/KWU, Erlangen, Germany
I. K. Abu-Shumays, Bettis Atomic Power Laboratory, West Mifflin, PA
R. Alcouffe, Los Alamos National Laboratory, Los Alamos, NM
J. E. Morel, Los Alamos National Laboratory, Los Alamos, NM
Y. Y. Azmy, Oak Ridge National Laboratory, Oak Ridge, TN
R. Abboud, Commonwealth Edison Co., Chicago, IL
D. Nigg, INEL, Idaho Falls, ID
R. Lawrence, IBM, Kingston, NY

External: (Cont'd.)

K. S. Smith, Studsvik USA, Idaho Falls, ID
C. B. Carrico, BARRA, Berkeley, CA (2)
M. Salvatores, CEA, Cadarache, France
P. J. Finck, CEA, Cadarache, France
R. Jacquemin, CEA, Cadarache, France
A. Gandini, ENEA, Casaccia, Italy
A. Dangelo, ENEA, Casaccia, Italy
J. P. West, EDF-DER, Clamart, France
T. Wu, General Electric, San Jose, CA
P. Ravetto, University of Turin, Turin, Italy
W. Gudowsky, Royal Institute of Technology, Stockholm, Sweden
J. J. Lautard, CEA, Saclay, France
R. Sanchez, CEA, Saclay, France
D. Biron, EDF-SEPTEN, Lyon, France
P. J. Collins, PNC, Tsuruga-Shi, Japan
L. Agee, EPRI, Palo Alto, CA
Y. A. Chao, Westinghouse Electric Corp., Pittsburgh, PA
N. Tsoulfanidis, Univ. of Missouri-Rolla, Rolla, MO
W. S. Yang, Chosun University, South Korea