NESTLE

FEW-GROUP NEUTRON DIFFUSION EQUATION SOLVER UTILZING THE NODAL EXPANSION METHOD FOR EIGENVALUE, ADJOINT, FIXED-SOURCE STEADY-STATE AND TRANSIENT PROBLEMS

Principal Investigator

Paul J. Turinsky

Research Assistants

Rifat M. K. Al-Chalabi

Philippe Engrand

Hisham N. Sarsour

Francois Xavier Faure

Weiping Guo

Published June 1994

Electric Power Research Center

North Carolina State University

Raleigh, NC 27695-7909

Prepared for the

U.S. Department of Energy

Idaho Operations Office

Under DOE Contract No. DE-AC07-761D01570

and

University of California

Los Alamos National Laboratory

Under Subcontract 9-X52-Z5141-1

DISCLAIMER

DISCLAIMER

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

Abstract

NESTLE is a FORTRAN77 code that solves the few-group neutron diffusion equation utilizing the Nodal Expansion Method (NEM). NESTLE can solve the eigenvalue (criticality); eigenvalue adjoint; external fixed-source steady-state; or external fixed-source or eigenvalue initiated transient problems. The code name NESTLE originates from the multi-problem solution capability, abbreviating **N**odal **E**igenvalue, **S**teady-state, **T**ransient, **Le** core **E**valuator. The eigenvalue problem allows criticality searches to be completed, and the external fixed-source steady-state problem can search to achieve a specified power level. Transient problems model delayed neutrons via precursor groups. Several core properties can be input as time dependent.

Two or four energy groups can be utilized, with all energy groups being thermal groups (*i.e.* upscatter exits) if desired. Core geometries modelled include Cartesian and Hexagonal. Three, two and one dimensional models can be utilized with various symmetries. The non-linear iterative strategy associated with the NEM method is employed. An advantage of the non-linear iterative strategy is that NESTLE can be utilized to solve either the nodal or Finite Difference Method representation of the few-group neutron diffusion equation.

Thermal-hyraulic feedback is modelled employing a Homogenous Equilibrium Mixture (HEM) model, allowing two-phase flow to be treated. However, only the continuity and energy equations for the coolant are solved, implying a constant pressure treatment. The slip is assumed to be one in the HEM model. A lumped parameter model is employed to determine the fuel temperature. Decay heat groups are used to model decay heat.

The thermal conditions predicted by the thermal-hyraulic model of the core are used to correct cross-sections for temperature and density effects. Cross-sections are parameterized by color, control rod state (*i.e.* in or out) and burnup, allowing fuel depletion to be modelled. Either a macroscopic or microscopic model may be employed. All cross-sections are expressed in terms of a Taylor's series expansion in coolant density, coolant temperature, effective fuel temperature,

and soluble poison number density.

Memory management is accomplished utilizing a container array to facilitate efficient memory allocation. In this manner various problems with different dimensionality can be executed without code re-compilation. To facilitate the understanding of coding, procedures are used extensively and an electronic dictionary program, NESTLE.DICT has been created to define the meaning of code variables.

Acknowledgments

The Principal Investigator wishes to acknowledge the assistance of several individuals on this project in addition to the research assistants indicated on the cover page. Special thanks goes to Jerry Judd of INEL, Mike Sweetman of NCSU, Tom Downar of Purdue University, and Teh-Chin Cheng of LANL for benchmarking the code. Conversations with Russ Mosteller of LANL are much appreciated. The help of Dave Kropaczek and Patrick VanVickle, both of NCSU, on 'roughing in' the Hexagonal-Z geometry NEM routines and preparing this report using FRAME, respectively, are acknowledged. Finally I would like to recognize the patience of this project's contract monitors, Dave Nigg of INEL and George Niederauer of LANL, for tolerating numerous delays in completion of tasks from scheduled dates. I hope the quality of the product was worth the delays!

Table of Contents

Listing of Tables

Listing of Figures

I Introduction

NESTLE is a FORTRAN77 code that solves the few-group neutron diffusion equation utilizing the Nodal Expansion Method (NEM). NESTLE can solve the eigenvalue (criticality); eigenvalue adjoint; external fixed-source steady-state; or external fixed-source or eigenvalue initiated transient problems. The code name NESTLE originates from the multi-problem solution capability, abbreviating **N**odal **E**igenvalue, **S**teady-state, **T**ransient, **Le** core **E**valuator. The eigenvalue problem allows criticality searches to be completed on one of the following variables: soluble boron, coolant inlet temperature, control rod position or core power level. The external fixed-source steady-state problem can also search on these same parameters, now in regard to achieving a specified power level.

Two or four energy groups can be utilized, with all groups being thermal groups (*i.e.* upscatter exits) if desired. Core geometries modelled include Cartesian and Hexagonal. Three, two and one dimensional models can be utilized. Various core symmetry options are available, including quarter, half and full core for Cartesian geometry and one-sixth, one-third and full core for Hexagonal geometry. Zero flux, non-reentrant current, reflective and cyclic boundary conditions are treated

The few-group neutron diffusion equation is spatially discretized utilizing the **N**odal **E**xpansion **M**ethod (NEM). Quartic or quadratic polynomial expansions for the transverse integrated fluxes are employed for Cartesian or Hexagonal geometries, respectively. Transverse leakage terms are represented by a quadratic polynomial or constant for Cartesian or Hexagonal geometry, respectively. **D**iscontinuity **F**actors (DFs) are utilized to correct for homogenization errors. Transient problems utilize a user specified number of delayed neutron precursor groups. Time dependent inputs include coolant inlet temperature and flow; soluble poison concentration, and control banks' positions. Time discretization is done in a fully implicit manner utilizing a first-order difference operator for the diffusion equation. The precursor equations are analytically

solved assuming the fission rate behaves linearly over a time-step.

Independent of problem type, an outer-inner iterative strategy is employed to solve the resulting matrix system. Outer iterations can employ Chebyshev acceleration and the Fixed Source Scaling Technique to accelerate convergence. Inner iterations employ either color line or point SOR iteration schemes, dependent upon problem geometry. Values of the energy group dependent optimum relaxation parameter and the number of inner iterations per outer iteration to achieve a specified L_2 relative error reduction are determined a priori. The non-linear iterative strategy associated with the NEM method is utilized. This has advantages in regard to reducing FLOP count and memory size requirements versus the more conventional linear iterative strategy utilized in the surface response formulation. In addition, by electing to not update the coupling coefficients in the nonlinear iterative strategy, the **F**inite **D**ifference **M**ethod (FDM) representation, utilizing the box scheme, of the few-group neutron diffusion equation results. The implication is that NESTLE can be utilized to solve either the nodal or FDM representation of the few-group neutron diffusion equation.

Thermal-hydraulic feedback is modelled employing a **H**omogenous **E**quilibrium **M**ixture (HEM) model, allowing two-phase flow to be treated. However, only the continuity and energy equations for the coolant are solved, implying a constant pressure treatment. The slip is assumed to be one in the HEM model. The fuel temperature is determined utilizing a lumped parameter model. The SETS method is used for the temporal treatment to overcome the material Courant limit on numerical stability. A conventional staggered mesh formulation is used in spatially discretizing the fluid's equations. Flow is assumed to be parallel to the axial direction within a closed channel. A user specified number of decay heat groups are used to model decay heat. Direct deposition in the coolant of fission energy is accounted for. Equation of State information is provided via polynomials, whose coefficients are provided as input. It should be recognized that the thermal-hydraulic model was developed with a pin-cell geometry as its basis. Adoption to

other geometries, such as appear in gas-cooled reactors, would likely require some source code modifications.

The thermal conditions predicted by the thermal-hydraulic model of the core are used to correct cross-sections for temperature and density effects. Cross-sections are parameterized by color, control rod state (*i.e.* in or out) and burnup, implying fuel burnup modelling capabilities exist. Either a macroscopic or microscopic fuel depletion model may be employed. A Predictor-Corrector formulation is used to solve the depletion equations. With the election of the microscopic option, depletion equations for the U^{234} through U^{236} and U^{238} through Pu^{242} depletion chains, two lumped fission product groups, and a simple burnable poison are solved and used in conjunction with burnup dependent microscopic cross-sections to construct the macroscopic cross-sections. The I-Xe and Pm-Sm chains are also modelled, with various options to determine their number densities (*i.e.* equilibrium, transient, peak Sm-no Xe, no Sm nor Xe, or frozen). All cross-sections are characterized in terms of a Taylor's series expansion in coolant density, coolant temperature, effective fuel temperature, and soluble poison number density. Taylor's series terms utilized (*e.g.* linear or quadratic in coolant density) are specified via input.

Output edits include predicted values of the key core attributes, such as power, flux, temperatures, isotopic number densities and burnup spatial distributions, in addition to documenting key input options and convergence behavior parameters. The output information is biased towards the sort of information a nuclear designer of a power reactor requires. A restart file is written, allowing restart for branch cases, re-initiation of core depletion, continuation of iterations towards a tighter convergence, or re-initiation of a transient.

Memory management is accomplished via a container array. Code determined container array pointers are used to facilitate problem specific memory allocation (*e.g.* trading off of spatial and energy detail within a fixed total memory size).

II Theoretical Foundations

II.1 Nodal Model - Cartesian Geometry

II.1.a Eigenvalue Problem

The following section describes the standard NEM formulation for the solution of the three-dimensional, Cartesian geometry, multi-group, eigenvalue neutron diffusion equation [1,2]. The principal characteristics of the polynomial nodal method are its quartic expansions of the one-dimensional transverse-integrated flux and quadratic leakage model for the transverse leakage.

Consider the general form of the steady-state multi-group neutron diffusion equation, written in standard form and with the group constants (*i.e.* properly weighted cross-sections and discontinuity factors) already available from a lattice physics calculation for $g = 1, 2,..., G$

$$\vec{\nabla} \cdot D_g \vec{\nabla} \phi_g + \Sigma_{tg} \phi_g = \sum_{g'=1}^{G} \Sigma_{sgg'} \phi_{g'} + \frac{\chi_g}{k} \sum_{g'=1}^{G} \nu_{g'} \Sigma_{fg'} \phi_{g'} \qquad (1)$$

where the dependence of each quantity on the spatial coordinate $\vec{r}$ has been suppressed, and,

D_g = diffusion coefficient [cm]

ϕ_g = neutron flux [$cm^{-2}sec^{-1}$]

Σ_{tg} = total macroscopic cross section [cm^{-1}]

$\Sigma_{sgg'}$ = group-to-group scattering cross section [cm^{-1}]

χ_g = fission neutrons yield

k = multiplication factor (*i.e.* critical eigenvalue)

ν_g = average number of neutrons created per fission

Σ_{fg} = macroscopic fission cross section [cm^{-1}]

As with most modern nodal methods, we begin by intergrating the multi-group neutron

diffusion equation over a material-centered spatial node which has homogenized properties. For Cartesian geometry we rewrite Eqn. (1) for the arbitrary spatial node l,

$$-D_g^l \frac{\partial^2}{\partial x^2}\phi_g^l(\vec{r}) - D_g^l \frac{\partial^2}{\partial y^2}\phi_g^l(\vec{r}) - D_g^l \frac{\partial^2}{\partial z^2}\phi_g^l(\vec{r}) + A_g^l \phi_g^l(\vec{r}) = Q_g^l(\vec{r}) \tag{2}$$

where, $g \in (1, G)$, and

$$(\vec{r}) \equiv (x, y, z) \in V^l = \Delta x \Delta y \Delta z \equiv \text{Volume of node } l$$

$$A_g^l = \Sigma_{tg}^l - \Sigma_{sgg}^l - \frac{x_g^l}{k} \nu_g \Sigma_{fg}^l$$

$$Q_g^l(\vec{r}) = \sum_{g' \neq g}^{G} Q_{gg'}^l \phi_{g'}^l(\vec{r}) = \sum_{g' \neq g}^{G} \Sigma_{sgg'}^l \phi_{g'}^l(\vec{r}) + \frac{x_g^l}{k} \sum_{g' \neq g}^{G} \nu_{g'} \Sigma_{fg'}^l \phi_{g'}^l(\vec{r})$$

For simplicity, in cases where redundant equations exist in all three directions, the illustrating equations will be only given in the x-direction. Using Fick's Law, which in the x-direction can be expressed as,

$$j_{gx}^l(\vec{r}) = -D_g^l \frac{\partial}{\partial x}\phi_g^l(\vec{r}) \tag{3}$$

where,

$$j_{gx}^l(\vec{r}) \equiv \text{x-component of the net neutron current}$$

allows Eqn.(2) to be rewritten as:

$$\frac{\partial}{\partial x} j_{gx}^{l}(\vec{r}) + \frac{\partial}{\partial y} j_{gy}^{l}(\vec{r}) + \frac{\partial}{\partial z}\, j_{gz}^{l}(\vec{r}) + A_g^l \phi_g^l(\vec{r}) \;=\; Q_g^l(\vec{r}) \tag{4}$$

Integration of Eqn. (4) over the volume of node l generates a local neutron balance equation in terms of the face-averaged net currents and the node volume average flux.

$$\frac{1}{\Delta x^l}(\bar{L}_{gx}^l) + \frac{1}{\Delta y^l}(\bar{L}_{gy}^l) + \frac{1}{\Delta z^l}(\bar{L}_{gz}^l) + A_g^l \bar{\phi}_g^l \;=\; \bar{Q}_g^l \tag{5}$$

where, assuming node l is centered around the coordinate's origin, the volume integrated quantities are defined below:

$$\bar{\phi}_g^l \;=\; \frac{1}{V^l} \int_{-\frac{\Delta x^l}{2}}^{\frac{\Delta x^l}{2}} \int_{-\frac{\Delta y^l}{2}}^{\frac{\Delta y^l}{2}} \int_{-\frac{\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} \phi_g^l(\vec{r})\, dxdydz \equiv \text{Node volume average flux}$$

$$\bar{Q}_g^l \;=\; \frac{1}{V^l} \int_{-\frac{\Delta x^l}{2}}^{\frac{\Delta x^l}{2}} \int_{-\frac{\Delta y^l}{2}}^{\frac{\Delta y^l}{2}} \int_{-\frac{\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} Q_g^l(\vec{r})\, dxdydz \equiv \text{Node volume average source}$$

and,

$$\frac{1}{\Delta x^l}\bar{L}_{gx}^l \;=\; \frac{1}{\Delta x^l}(\bar{J}_{gx+}^l - \bar{J}_{gx-}^l) \;=\; \frac{1}{V^l} \int_{-\frac{\Delta x^l}{2}}^{\frac{\Delta x^l}{2}} \int_{-\frac{\Delta y^l}{2}}^{\frac{\Delta y^l}{2}} \int_{-\frac{\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} \frac{\partial}{\partial x} j_{gx}^{l}(\vec{r})\, dxdydz$$

where,

$$\bar{J}^l_{gx\pm} \equiv \text{Average x-directed net current on node faces} \pm \frac{\Delta x^l}{2}$$

Eqn. (5) is known as the nodal balance equation. Now for the neutron diffusion equation written in this form, in order to obtain the spatial neutron flux distribution, one must devise some relationship between the node average flux and the face-averaged net (surface) currents. It is the equations used to compute the surface currents in Eqn. (5) which distinguish one nodal formulation from another. In NEM, the widely used method of transverse-integration is used, where the three-dimensional diffusion equation is integrated over the two directions transverse to each axis. This generates three one-dimensional equations, one for each direction in Cartesian coordinates, of the following form,

$$\frac{d}{dx}\bar{j}^l_{gx}(x) + A^l_g \bar{\phi}^l_{gx}(x) = \bar{Q}^l_{gx}(x) - \frac{1}{\Delta y^l}\bar{L}^l_{gy}(x) - \frac{1}{\Delta z^l}\bar{L}^l_{gz}(x) \tag{6}$$

where,

$$\bar{L}^l_{gy}(x) = \frac{1}{\Delta z^l}\int_{-\frac{\Delta z^l}{2}}^{\frac{\Delta z^l}{2}}\int_{-\frac{\Delta y^l}{2}}^{\frac{\Delta y^l}{2}} \frac{\partial}{\partial y} j^l_{gy}(\hat{r})\, dydz \equiv \text{Average y-direction transverse leakage}$$

and,

$$\bar{L}^l_{gz}(x) = \frac{1}{\Delta y^l} \int_{-\frac{\Delta y^l}{2}}^{\frac{\Delta y^l}{2}} \int_{-\frac{\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} \frac{\partial}{\partial z} j^l_{gz}(\vec{r})\, dz dy \equiv \text{Average z-direction transverse leakage}$$

In NEM, the one-dimensional averaged flux that appears in Eqn. (6), is expanded as a general polynomial,

$$\bar{\phi}^l_{gx}(x) = \bar{\phi}^l_g + \sum_{n=1}^{N} a^l_{gxn}\, f_n(x) \tag{7}$$

where $\bar{\phi}^l_g$ is the node average flux, implying for Eqn. (7) to be true that $f_n(x)$ must be chosen such that the basis functions satisfy

$$\int_{-\frac{\Delta x^l}{2}}^{\frac{\Delta x^l}{2}} f_n(x)\, dx = 0 \text{ for } n = 1, ..., N \tag{8}$$

Note that for quartic NEM, the method used in NESTLE, the summation extends to $N = 4$. The first four basis functions in NEM can be expressed as follows [1],

$$\overbrace{(\frac{x}{\Delta x^l})}^{f_1};\quad \overbrace{3(\frac{x}{\Delta x^l})^2 - \frac{1}{4}}^{f_2};\quad \overbrace{(\frac{x}{\Delta x^l})^3 - \frac{1}{4}(\frac{x}{\Delta x^l})}^{f_3};\quad \overbrace{(\frac{x}{\Delta x^l})^4 - \frac{3}{10}(\frac{x}{\Delta x^l})^2 + \frac{1}{80}}^{f_4} \tag{9}$$

which can be shown to also satisfy the following,

$$f_n(\pm\frac{\Delta x^l}{2}) = 0 \qquad \text{for} \qquad n = 3, 4 \tag{10}$$

At this point it is appropriate to consider the elementary concept of accounting for the total number of equations and that of unknowns. For a three-dimensional Cartesian geometry, the node

average and N expansion coefficients in each direction appear per node per energy group, implying a total of $3N+1$ equations are required. The nodal balance equation, Eqn. (5), provides one equation, where now Eqns. (3) and (7) are used to eliminate face-averaged net currents from this equation. Surface current and flux continuity provide 6 more equations per node per energy group. So for N=2, there would be an equal number of equations and unknowns without any further development. However, for N= 4, two additional unknowns are introduced for each direction per node per energy group. This is addressed by using a weighted residual scheme [3] applied to Eqn. (6), which in essence provides the additional equations (referred to as the moment equations) needed,

$$< \omega_n(x), \frac{d}{dx} \bar{j}^l_{gx}(x) > -A^l_g \bar{\phi}^l_{gxn} = \bar{Q}^l_{gxn} - \frac{1}{\Delta y^l} \bar{L}^l_{gyxn} - \frac{1}{\Delta z^l} \bar{L}^l_{gzxn} \tag{11}$$

where the two weighting functions for n = 1,2 are chosen to be the same as the basic functions, namely $\omega_n(x) = f_n(x)$, as those used in the one-dimensional flux expansion[1]. Here, the first and second (actually linear combination of zeroth and second) moments of the flux, source, and leakage for each group g are defined by,

$$\underbrace{\bar{\phi}^l_{gxn}}_{< \omega_n(x), \bar{\phi}^l_{gx}(x) >} \quad \underbrace{\bar{Q}^l_{gxn}}_{< \omega_n(x), \bar{Q}^l_{gx}(x) >} \quad \underbrace{\bar{L}^l_{gyxn}}_{< \omega_n(x), \bar{L}^l_{gy}(x) >} \quad \underbrace{\bar{L}^l_{gzxn}}_{< \omega_n(x), \bar{L}^l_{gz}(x) >}$$

The first term in Eqn. (11) is evaluated by using Eqns. (3) and (7) and the definition of the expansion coefficients, and completing the integration (*i.e.* inner product) analytically.

One last point which needs to be addressed before Eqn. (11) can be solved are the transverse leakage terms appearing on the right hand side. Their spatial dependency is unknown,

1. This constitutes a *moments weighting* scheme; if one uses $\omega_n(x) = f_{n+2}(z)$ for n = 1,2 it is known as *Galerkin weighting*. Numerical experiments favor *moments weighting*

so their "shape" must be approximated. The most popular approximation in NEM is the quadratic transverse leakage approximation. For example, the x-direction spatial dependence of the y-direction transverse leakage is approximated by,

$$\bar{L}^l_{gy}(x) \cong \bar{L}^l_{gy} + \rho^l_{gy1} f_1(x) + \rho^l_{gy2} f_2(x) \tag{12}$$

where $\bar{L}^l_g$ is the average y-directed leakage in node l, and the coefficients ρ^l_{gy1} and ρ^l_{gy2} can be expressed in terms of average y-directed leakages of the two nearest-neighbor nodes along the x-direction (*i.e.* nodes l-1 and l+1) so as to preserve the node average leakages of these three nodes. The quadratic expansion coefficients can be shown to be given by,

$$\rho^l_{gy1} = g^l(\Delta x^l)\,[\,(\bar{L}^{l+1}_g - \bar{L}^l_g)\,(\Delta x^l + 2\Delta x^{l-1})\,(\Delta x^l + \Delta x^{l-1}) + (\bar{L}^l_{gy} - \bar{L}^{l-1}_{gy})\,(\Delta x^l + 2\Delta x^{l+1})\,(\Delta x^l + \Delta x^{l+1})\,] \tag{13}$$

$$\rho^l_{gy2} = g^l(\Delta x^l)^2\,[\,(\bar{L}^{l+1}_{gy} - \bar{L}^l_{gy})\,(\Delta x^l + \Delta x^{l-1}) + (\bar{L}^{l-1}_{gy} - \bar{L}^l_{gy})\,(\Delta x^l + \Delta x^{l+1})\,] \tag{14}$$

where,

$$g^l = [\,(\Delta x^l + \Delta x^{l+1})\,(\Delta x^l + \Delta x^{l-1})\,(\Delta x^{l-1} + \Delta x^l + \Delta x^{l+1})\,]^{-1} \tag{15}$$

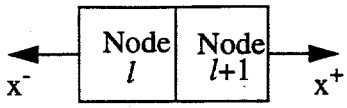
II.1.b Non-Linear Iterative Strategy

The most common manner of solving the matrix system associated with NEM is the response-matrix formulation. To minimize computer run time and memory requirements, and to facilitate the capability to solve either the NEM or Finite Difference Method (FDM) formulation, the non-linear iterative strategy is employed in NESTLE. This technique was developed by Smith [4, 5, 6] and successfully implemented into the Studsvik QPANDA and SIMULATE code packages. The documentation available on this technique is scarce, but it turns out to be rather simplistic and almost trivial to implement in a FDM code which utilizes the box-scheme (*i.e.* material-centered).

The basic idea is applicable to the standard FDM solution algorithm of the multi-group diffusion equation. Solving the FDM based equation utilizing an outer-inner iterative strategy, every ΔN_0 outer iterations (where ΔN_0 is somewhat arbitrary but can be optimized) the so-called "two-node problem" calculation (a spatially-decoupled NEM calculation spanning two adjoining nodes) is performed for every interface (for all nodes and in all directions) to provide an improved estimate of the net surface current at that particular interface. Subsequently, the NEM estimated net surface currents are used to update (*i.e.* change) the original FDM diffusion coupling coefficients. Outer iterations of the FDM based equation are than continued utilizing the updated FDM coupling coefficients for ΔN_0 outer iterations. The entire process is then repeated. This procedure of updating the FDM couplings is a convergent technique which progressively forces the FDM equation to yield the higher-order NEM predicted values of the net surface currents while satisfying the nodal balance Eqn. (5), thus yielding the NEM results for the node-average flux and fundamental mode eigenvalue. The advantages of this technique come in many forms; the storage requirements are minimal because the two-node problem arrays are re-usable (disposable) at each interface, the rate of convergence is nearly comparable to that of the base FDM algorithm being used, the number of iteratively determined unknowns is reduced by a factor

of 6 (node flux vs. partial surface current), and the simplicity of the algorithm and ease of implementation, compared to any other nodal technique, is far superior.

The two-node problem produces an 8G X 8G linear system of equations which can be constructed by applying the standard NEM relations to two adjoining nodes. For simplicity, consider two arbitrary adjoining nodes in the x-direction. Denote these notes as l and l+1:



Substitution of the one-dimensional expansion, Equation (7), into Fick's law yields expressions for the average x-direction net surface currents at the left(-) and right(+) interfaces of node l,

$$\bar{J}^{l}_{gx\pm} \equiv \frac{-D^{l}_{g}}{\Delta x^{l}}\left[a^{l}_{gx1} \pm 3a^{l}_{gx2} + \frac{1}{2}a^{l}_{gx3} \pm \frac{1}{5}a^{l}_{gx4}\right] \tag{16}$$

Now, assume the node average flux, criticality constant, and all transverse direction terms are known from a previous iteration; then, the total number of unknowns associated with the x-direction two node problem is 8G, which corresponds to the 4 expansion coefficients/group/node (x) G groups (x) two nodes. The 8G constraint equations are obtained as follows. We begin with the substitution of Eqn. (16) into the nodal balance equation for node l, to yield the zeroth moment constraints (G equations/node),

$$\frac{-D^{l}_{g}}{\Delta x^{l}}\left[6a^{l}_{gx2} + \frac{2}{5}a^{l}_{gx4}\right] = -\frac{1}{\Delta y^{l}}\bar{L}^{l}_{gy} - \frac{1}{\Delta z^{l}}\bar{L}^{l}_{gz} - A^{l}_{g}\bar{\phi}^{l}_{g} + \sum_{g' \neq g}^{G} Q^{l}_{gg'}\bar{\phi}^{l}_{g'} \tag{17}$$

A similar substitution into the moment-weighted equation, Equation (11), yields the first and second moment constraints (2G equations/node),

$$\left[\frac{60}{\Delta x^{l}}\frac{D^{l}_{g}}{\Delta x^{l}} + A^{l}_{g}\right]a^{l}_{gx3} - \sum_{g' \neq g}^{G} Q^{l}_{gg'}a^{l}_{g'x3} - 10A^{l}_{g}a^{l}_{gx1} + 10\sum_{g' \neq g}^{G} Q^{l}_{gg'}a^{l}_{g'x1} = 10\left(\frac{1}{\Delta y^{l}}\rho^{l}_{gy1} + \frac{1}{\Delta z^{l}}\rho^{l}_{gz1}\right) \tag{18}$$

$$\left[\frac{140}{\Delta x^l}\frac{D_g^l}{\Delta x^l}+A_g^l\right]a_{gx4}^l-\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'x4}^l-35A_g^l a_{gx2}^l+35\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'x2}^l=35\left(\frac{1}{\Delta y^l}\rho_{gy2}^l+\frac{1}{\Delta z^l}\rho_{gz2}^l\right) \tag{19}$$

Similar equations can be written for node l+1, producing a total of 6G equations. The continuity of net surface current constraints at the interface (G equations) are obtained by using Equation (16) at the adjoining interface of the two nodes,

$$\frac{-D_g^l}{\Delta x^l}\left[a_{gx1}^l+3a_{gx2}^l+\frac{a_{gx3}^l}{2}+\frac{a_{gx4}^l}{5}\right]=\frac{-D_g^l}{\Delta x^{l+1}}\left[a_{gx1}^{l+1}-3a_{gx2}^{l+1}+\frac{a_{gx3}^{l+1}}{2}-\frac{a_{gx4}^{l+1}}{5}\right] \tag{20}$$

Last, the continuity (or discontinuity) of surface-averaged flux constraints (G equations) are obtained by equating the surface-averaged fluxes of the two adjoining nodes by using Equation (7),

$$d_{gx+}^l\left[\phi_g^l+\frac{a_{gx1}^l}{2}+\frac{a_{gx2}^l}{2}\right]=d_{gx-}^{l+1}\left[\phi_g^{l+1}-\frac{a_{gx1}^{l+1}}{2}+\frac{a_{gx2}^{l+1}}{2}\right] \tag{21}$$

where $d_{gx\pm}^l$ and $d_{gx\pm}^{l+1}$ are the Discontinuity Factors (DFs) obtained from lattice physics calculations. Do note that continuity conditions are never imposed on the outside surfaces of the two-node problem, since the two-node problem is deliberately formulated to be spatially decoupled. Continuity is assured in the formulation of the FDM based equations.

Equations (7) through (21) constitute the 8G system of equations needed to be solved at each interface. This matrix system, after taking advantage of its reducability and by noting that the even-moment expansion coefficients don't change whether the node is on the left or right of a two-node problem, can be reduced to smaller systems which can be solved quite efficiently [7]. The following table illustrates this more efficient arrangement of unknowns for the case of G=2.

Table 1: Non zero entries in the 16 by 16 two-node NEM problem

Equation	Grp	Nod	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
0th Moment	1	*l*	x		x													
0th Moment	2	*l*		x		x												
2nd Moment	1	*l*	x	x	x	x												
2nd Moment	2	*l*	x	x	x	x												
0th Moment	1	*l*+1					x		x									
0th Moment	2	*l*+1						x		x								
2nd Moment	1	*l*+1					x	x	x	x								
2nd Moment	2	*l*+1					x	x	x	x								
1st Moment	1	*l*									x	x	x	x				
1st Moment	2	*l*									x	x	x	x				
1st Moment	1	*l*+1													x	x	x	x
1st Moment	2	*l*+1													x	x	x	x
Cur Con	1		x		x		x		x		x		x		x		x	
Cur Con	2			x		x		x		x		x		x		x		x
Flx Dis	1		x				x				x				x			
Flx Dis	2			x				x				x				x		

UNKNOWN	NODE	GROUP	EXP. COEF.*
a	*l*	1	2
b	*l*	2	2
c	*l*	1	4
d	*l*	2	4
e	*l*+1	1	2
f	*l*+1	2	2
g	*l*+1	1	4
h	*l*+1	2	4
i	*l*	1	1
j	*l*	2	1
k	*l*	1	3
l	*l*	2	3
m	*l*+1	1	1
n	*l*+1	2	1
o	*l*+1	1	3
p	*l*+1	2	3

*Refers to order of polynomial that transverse

integrated flux expansion coefficient is associated with.

In NESTLE, the two-node problems are solved by utilizing the analytic solution to the 8G X 8G matrix system. This was accomplished by employing symbolic manipulator software to produce the FORTRAN code segment used in NESTLE. This approach is computationally more efficient

than utilizing a direct matrix solver (*e.g.* LU decomposition); however, it limits the values of G to those directly programmed for. Also note that on boundaries special treatments of the two-node problems are required. Depending upon the specified boundary condition (BC), one-node problems may originate (*e.g.* zero flux BC), or on interior axis geometry unfolding may be required to create a two-node problem (*e.g.* cyclic BC).

Solutions of the two-node problems provide NEM evaluated values of the currents on all surfaces for specified values of the node average fluxes [recall they were assumed known in solving the two-node problems]. To correct the FDM based expression for the surface current, the following approach is utilized. The coupling coefficient update to the FDM equation can be implemented by simply expressing the FDM net surface current at the $x+$ face of node l as follows,

$$\bar{J}_{gx+}^{l,\,FDM} = \frac{D_{gx+}^{l,\,FDM}}{\frac{\Delta x^l + \Delta x^{l+1}}{2}} [\bar{\phi}_g^{l+1} - \bar{\phi}_g^l] - \frac{\tilde{D}_{gx+}^{l,\,NEM}}{\frac{\Delta x^l + \Delta x^{l+1}}{2}} [\bar{\phi}_g^{l+1} + \bar{\phi}_g^l] \tag{22}$$

The first term on the RHS is the normal FDM approximation for a box scheme, where $D_{gx+}^{l,\,FDM}$ is the actual FDM diffusion coupling coefficient between nodes l and l+1,

$$D_{gx+}^{l,\,FDM} = \frac{D_g^l D_g^{l+1} (\Delta x^l + \Delta x^{l+1})}{D_g^l \Delta x^l + D_g^{l+1} \Delta x^{l+1}} \tag{23}$$

The second term on the RHS represents the nonlinear NEM correction applied to the FDM scheme. The (+) sign between the flux values in the second term of Equation (22) is purposely there to improve the convergence behavior of the nonlinear iterative method [8]. Note that if $\tilde{D}_{gx+}^{l,\,NEM}$ is zero, which it initially is in NESTLE's implementation, then Equation (22) corresponds to the standard FDM definition of the net surface current. This is the basis for the FDM option within NESTLE, where now two-node problem solves and coupling coefficients

updates are never completed. The value of $\tilde{D}_{gx+}^{l,NEM}$ is determined by setting Equation (22) equal to the NEM two-node predicted surface current value, using the associated node average flux values in Equation (22) and solving for this quantity.

Summarizing, to apply a NEM update after ΔN_0 outer iterations of the FDM routine, one solves the two-node problem at a given interface, then (with the expansion coefficients known for that interface) one calculates the NEM estimate of the net surface current using Equation (16) Finally, one equates this result to Equation (22), and solves for the value of $\tilde{D}_{gx+}^{l,NEM}$ which will be used in the subsequent set of FDM iterations.

II.2 Outer-Inner Solution Method for FDM Equations

The only large matrix that requires solution for the non-linear iterative method is the FDM representation of the multi-group diffusion equation. Much work has been done on formulating, understanding and implementing the iterative solution of this large, sparse matrix system. NESTLE takes advantage of this wealth of knowledge in its iterative solution implementation, utilizing an outer-inner iterative strategy.

The "Outer-Inner Method" refers to outer iterations to update the fission source term and inner iteration to approximately solve the resulting fixed source problem. The outer iterations correspond to a "Power Method." This method can be applied to both Fixed Source Problems [FSP] and the Associated Eigenvalue Problem [AEVP]. Shortly it will be shown that both the fixed source steady-state and transient problems are representable as FSP in NESTLE's formulation. Although the AEVP involves additional calculations for the eigenvalue, basically the iteration schemes for both problems are similar. We will discuss the AEVP first.

Returning to Equation (5), the FDM representation of this equation in three-dimensional Cartesian geometry within homogenous mode l can be expressed as follows:

$$\sum_{l'=1}^{L} C_g^{l,l'} \bar{\phi}_g^{l'} + A_g^l \bar{\phi}_g^l = \bar{Q}_g^l \tag{24}$$

where the non-zero values of the coupling coefficients $\{C_g^{l,l'}\}$ are obtained via Equations(22) and (23) and L denotes the total number of nodes. Substituting in the definitions for A_g^l and $\bar{Q}_g^l$ into Equation (24) and rearranging terms we obtain

$$\sum_{l' \neq l}^{L} C_g^{l,l'} \bar{\phi}_g^{l'} + (\Sigma_{t_g}^l - \Sigma_{s_{gg}}^l + C_g^{l,l}) \bar{\phi}_g^l - \sum_{g' \neq g}^{G} \Sigma_{s_{gg'}}^l \bar{\phi}_{g'}^l = \frac{x_g^l}{k} \sum_{g'=1}^{G} \upsilon_{g'} \Sigma_{fg'}^l \bar{\phi}_{g'}^l \tag{25}$$

This equation can be written in terms of matrix notation spanning the spatial domain as

$$\overline{\overline{A}}_g \bar{\phi}_g - \sum_{g' \neq g}^{G} \overline{\overline{\Sigma}}_{s_{gg'}} \bar{\phi}_{g'} = \frac{1}{k} \overline{\overline{x}}_g \sum_{g'=1}^{G} \overline{\overline{\upsilon_{g'} \Sigma^l_{fg'}}} \bar{\phi}_{g'} \tag{26}$$

where the "bar" over the node average flux value now denotes a column vector. Matrix $\overline{\overline{A}}_g$ has a seven-banded matrix structure for three-dimensional Cartesian geometry. In turn, the G (L X L) matrix systems expressed by Equation (26) can be collected to write the following single (GL X GL) matrix system.

$$\overline{\overline{A}} \bar{\phi} = \frac{1}{k} \overline{\overline{F}} \bar{\phi} \tag{27}$$

The matrix $\overline{\overline{A}}$ is block lower triangular in structure for that portion applicable to the fast groups.

The outer-inner iteration process is summarized as follows: For the AEVP specified by Equation (27), given an arbitrary initial vector $\bar{\phi}^{(0)}$, the outer iterations generate successive estimates for the flux vector $\bar{\phi}$ by the process

$$\bar{\phi}^{(q)} = \frac{1}{k^{(q-1)}} \overline{\overline{A}}^{-1} (\overline{\overline{F}} \bar{\phi}^{q-1}) \tag{28}$$

where how the criticality constant (*i.e.* eigenvalue) is updated will be discussed later. The iterative matrix associated with the outer iterations is

$$\overline{\overline{Q}} = \overline{\overline{A}}^{-1} \overline{\overline{F}} \tag{29}$$

The properties of the iterative matrix $\overline{\overline{Q}}$ has a significant role in determining the convergence rate of the power iterations [9, 10].

In solving Equation (28), advantage is taken of the structure of the $\overline{\overline{A}}$ matrix. For the fast groups, solving from low to high energy group number results in energy group decoupling. This implies that we may solve a system of linear equations of the form

$$\overline{\overline{A}}_g \bar{\phi}_g^{(q)} = \bar{S}_g^{(q)} \tag{30}$$

where,

$$\bar{S}_g^{(q)} = \sum_{g' \neq g}^{G} \bar{\bar{\Sigma}}_{sgg'} \bar{\phi}_{g'}^{(q)} + \frac{1}{k^{(q-1)}} \bar{\bar{\chi}}_g \sum_{g'=1}^{G} \overline{\overline{\nu_{g'} \Sigma_{fg'}}} \bar{\phi}_{g'}^{(q-1)} \tag{31}$$

For the thermal groups, NESTLE assumes the group fluxes for all other thermal groups except the one being updated are known. This produces energy group decoupling, allowing Equation (30) to be utilized. So called "scattering" iterations are then completed after all thermal groups' fluxes are updated. Stationary acceleration is employed to accelerate convergence of the scattering iterations.

II.2.a Inner Iteration Acceleration

To solve Equation (30) we introduce the inner iterations. In this work we employ a Multi-Color Point or Line SOR Method, depending upon problem geometry, for the inner iterations. Specifically, a Red-Black Point or Line SOR method is used in NESTLE for two or three-dimensional Cartesian geometry, respectively. For one-dimensional Cartesian geometry, a direct matrix solve is utilized since the group-wise A matrix is triangular allowing employment of Gaussian elimination.

Mathematically, this approach is a multi-spliting method and can be expressed as follows.

$$\bar{\phi} = \oplus \bar{\phi}_p \text{ where vector } \bar{\phi}_p \text{ spans nodes of color "p"}$$

$$\bar{\phi}_p^{(m+1)} = \bar{\bar{B}}_p^{-1} \left[\bar{S} + \sum_{p'=1}^{p-1} \bar{\bar{C}}_{pp'} \bar{\phi}_{p'}^{(m+1)} + \sum_{p'=p+1}^{P} C_{pp'} \bar{\phi}_{p'}^{(m)} \right] \quad for \quad p = 1,2,...,P \tag{32}$$

where,

$$\bar{\bar{A}} = \otimes \bar{\bar{A}}_p \text{ and non-square matrix } \bar{\bar{A}}_p \text{ equals rows of } \bar{\bar{A}} \text{ that span nodes of color "p"}$$

$$\overline{\overline{A}}_p = \overline{\overline{B}}_p - \sum_{p' \neq p}^{P} \overline{\overline{C}}_{pp'} \qquad for \qquad p = 1,2,...,P \tag{33}$$

and

$$\underline{\phi}_p^{(m+1)} = \underline{\phi}_p^{(m)} + \omega (\bar{\phi}_p^{(m+1)} - \underline{\phi}_p^{(m)}) \tag{34}$$

Note that the group g and outer iteration count (q) indices have been suppressed for clarity in the above equations. The matrix $\overline{\overline{B}}_p$ is square and has either a diagonal structure for the point scheme or block diagonal structure composed of tridiagonal blocks for the line scheme. This implies that the action of $\overline{\overline{B}}_p^{-1}$ indicated in Equations (32) is simple to evaluate. A total of ΔN_I inner iterations per outer iterations are completed, this value determined such that the specified relative error reduction from the 0^{th} iterative error for the inner iterations is achieved.

To a priori determine the value of the optimum relaxation parameter, ω and ΔN_I [which are energy group dependent but dependence notation has been surpressed], it is assumed that the iterative matrix associated with this inner iterative method is symmetrizable. This is not true since the NEM corrections to the FDM coupling coefficients invalidate symmetry; however, these corrections have been found to be relatively small so the symmetrizable assumption is acceptable. Making this assumption, we can express ω in terms of the spectral radius of the associated Guass-Seidel iteration matrix, $\rho(\overline{\overline{L}}^{G\text{-}S})$, as follows,

$$\omega = \frac{2}{1 + [1 - \rho(L^{G\text{-}S})]^{1/2}} \tag{35}$$

Clearly $\overline{\overline{L}}^{G\text{-}S} = \overline{\overline{L}}^{SOR}(\omega)$ with $\omega = 1$. Therefore, calculation of the spectral radius of the associated Guass-Seidel iterative matrix is the heart of this procedure. The following summarizes the details of the computational procedure used in NESTLE to obtain an estimate of the value of ω, which is based upon the DIF3D methodology [10]. These steps are completed for each energy

group.

Step 1. Starting with an arbitrary non-negative initial guess vector $\bar{x}^{(0)}$, complete at least ten Gauss-Seidel iterations in solving the following equation.

$$\bar{\bar{A}}\bar{x} = \bar{0}$$

Step 2. Following each iteration with $m > 10$, estimate the upper and lower bounds of the spectral radii using the following equations.

$$\lambda^{(m)} \equiv \frac{\langle \bar{x}^{(m)}, \bar{x}^{(m)} \rangle}{\langle \bar{x}^{(m)}, \bar{x}^{(m-1)} \rangle}$$

$$\bar{\lambda}^{(m)} \equiv MAX_i \left| \frac{x_i^{(m)}}{x_i^{(m-1)}} \right|$$

$$\underline{\lambda}^{(m)} \equiv MIN_i \left| \frac{x_i^{(m)}}{x_i^{(m-1)}} \right|$$

Compute the corresponding relaxation factors given by

$$\omega^{(m)} \equiv \frac{2}{1+[1-\lambda^{(m)}]^{1/2}}$$

$$\overline{\omega}^{(m)} \equiv \frac{2}{1+[1-\overline{\lambda}^{(m)}]^{1/2}}$$

$$\underline{\omega}^{(m)} \equiv \frac{2}{1+[1-\underline{\lambda}^{(m)}]^{1/2}}$$

Step 3. Terminate iteration when either

$$\left|\overline{\omega}^{(m)} - \underline{\omega}^{(m)}\right| < \frac{2-\omega^{(m)}}{5}$$

or m equals a specified upper limit [10,11]. The optimum factor ω is then set to $\omega^{(m)}$. This test forces tighter convergence of ω when $\rho\,(L^{G\text{-}S})$ is close to unity to ensure the required numerical accuracy is achieved.

Step 4. Determine the number of inner iterations required for each outer iteration ΔN_I, such that

the value of ΔN_I satisfies the following equation:

$$\left\| (\bar{\bar{L}}^{SOR}(\omega))^{\Delta N_{I-1}} \cdot \bar{\bar{L}}^{G-S} \right\| = [t^2_{2\Delta N_I - 1} + t^2_{2\Delta N_I}]^{1/2} \leq \varepsilon_{in}$$

where

$$t_{\Delta N_I} = [\omega - 1]^{\frac{\Delta N_I - 1}{2}} \left[\rho(\bar{\bar{L}}^{G-S})\right]^{1/2} \left[1 + (\Delta N_I - 1)\,(1 - \rho(\bar{\bar{L}}^{G\text{-}S}))^{1/2}\right]$$

and ε_{in} denotes the desired relative error reduction from the initial iteration to the end of ΔN_I-th iteration. It is suggested that a very small number for ε_{in} not be used since it may force excessive inner iterations [10].

The advantages of these accelerations strategies are clear. The automated determination of the optimum overrelaxation factors relieves users of the burden of the trial and error manner of specifying optimum parameters for a large class of reactor models. In addition, substantial computational time can be saved since the need to check the convergence of inner iterations has been removed by using a fixed number of predetermined inner iterations for each energy group.

The outer iterations defined by Equation (28) are slow to converge, since the dominance ratio of the iterative matrix, Equation (29), is close to one. Two complementary acceleration techniques are utilized in NESTLE to accelerate the outer iterations of the AEVP.

II.2.b Outer Iteration Acceleration

The outer iterations for the AEVP are accelerated by using linear combinations of the previous iterative vectors as now described. The Chebyshev polynomials are used to obtain the best linear combinations when there is no knowledge of higher eigenvalues [12]. The method implemented is the Chebyshev Semi-Iterative method [9,10,11,13]. In this method, the error vector associated with the acceleration method is expressed in terms of a linear combination of the error vectors of the underlying interactive method. Acceleration of the iteration is achieved by minimizing the error vector by appropriate selection of the expansion coefficients, which is determined to be those associated with Chebyshev polynomials. Further details of the mathematical background of this method can be found in the related references [9,10].

Since the rate of convergence in the AEVP is dependent on the dominance ratio $\sigma(\bar{\bar{Q}})$, the Chebyshev acceleration method detailed in Refs. [9,10,11,13] can therefore be applied to iterations,

$$\bar{\phi}^{(q)} = \frac{1}{k^{(q-1)}} \bar{\bar{Q}} \bar{\phi}^{(q-1)} \tag{36}$$

provided that a suitable estimate of $\sigma(\bar{\bar{Q}})$ is obtained. NESTLE follows the DIF3D approach to solve the AEVP in which we accelerate the fission source $\bar{\Psi}$ [13], where $\bar{\Psi}$ is defined as

$$\bar{\Psi} = \sum_{g'=1} \overline{\overline{\nu_{g'} \Sigma_{fg'}}} \bar{\phi}_{g'} \tag{37}$$

The accelerated iterative procedure can then be expressed as follows:

$$\bar{\Psi}^{(n^{*}+p)} = \frac{1}{k^{(n^{*}+p-1)}} \bar{\bar{Q}} \bar{\Psi}^{(n^{*}+p-1)} \tag{38}$$

where

$$\bar{\Psi}^{(n^{*}+p)} = \bar{\Psi}^{(n^{*}+p-1)} + \alpha_p \left[\bar{\Psi}^{(n^{*}+p)} - \bar{\Psi}^{(n^{*}+p-1)} \right] + \beta_p \left[\bar{\Psi}^{(n^{*}+p-1)} - \bar{\Psi}^{(n^{*}+p-2)} \right] \tag{39}$$

and

$$k^{(n^*+p)} = k^{(n^*+p-1)} \left(\frac{\left\| \overline{\Psi}^{(n^*+p)} \right\|_2^2}{(\overline{\Psi}^{(n^*+p)}, \overline{\Psi}^{(n^*+p-1)})} \right)$$

$$\alpha_1 = \frac{2}{2 - \sigma(\overline{\overline{Q}})}$$

$$\beta_1 = 0$$

$$\alpha_p = \frac{4}{\sigma(Q)} \left(\frac{\cosh[(p-1)\gamma]}{\cosh[p\gamma]} \right)$$

$$\beta_p = \left(1 - \frac{\sigma(\overline{\overline{Q}})}{2} \right) \alpha_p - 1$$

$$\gamma = \cosh^{-1} \left(\frac{2}{\sigma(\overline{\overline{Q}})} - 1 \right)$$

n^* = outer iteration index where acceleration begins

and p denotes the successive fission source iterations employed $(p \geq 1)$ within a Chebyshev cycle (*i.e.* since last updating the estimate of $\sigma(\overline{\overline{Q}})$). Note the dominance ratio $\sigma(\overline{\overline{Q}})$ needs to be estimated in order for the scheme to work. This is accomplished using the procedure implemented in DIF3D [10] as now outlined.

Since an accurate estimate of $\sigma(\overline{\overline{Q}})$ is not known when the outer iterations are commenced, a "boot-strap" process is required. By performing a limited number of power iterations, a reasonable initial estimate of $\sigma(\overline{\overline{Q}})$ is obtained. Only when all but the first overtone mode are essentially damped out, high-order cycles based on accurate estimates of $\sigma(\overline{\overline{Q}})$ are utilized [10,14]. More precisely, the algorithm can be described in terms of four basic steps:

Step 1. A minimum of three power iterations are performed initially. The first Chebyshev

acceleration cycle is begun on outer iteration $(n^* + 1)$, where $(n^* + 1)$ is the smallest integer such that $n^* \geq 3$ for which the dominance ratio estimate, $\hat{\sigma}$ satisfies the following criterion:

$$0.4 \leq \hat{\sigma} \leq 1.0$$

where

$$\hat{\sigma} = \left\{ \frac{\langle \bar{R}^{(n^*)}, \bar{R}^{(n^*)} \rangle}{\langle \bar{R}^{(n^*-1)}, \bar{R}^{(n^*-1)} \rangle} \right\}^{1/2}$$

$$\bar{R}^{(n^*)} \equiv \bar{\Psi}^{(n^*)} - \bar{\tilde{\Psi}}^{(n^*-1)}$$

Step 2. Using $\hat{\sigma}$ as the dominance ratio estimate for $\sigma(\bar{\bar{Q}})$, the accelerated iterative sequence given by Equations (38) and (39) is carried out for iterations $(n^* + p)$ with $p \geq 1$. At first low degree polynomials are applied repeatedly with estimates of the dominance ratio being updated continuously according to

$$\hat{\sigma}' = \frac{\hat{\sigma}}{2} \cosh\left(\left[\frac{\cosh^{-1}(\gamma)}{p-1} \right] + 1 \right)$$

where

$$\gamma = C_{p-1}\left(\frac{2-\hat{\sigma}}{\hat{\sigma}}\right)E_{n^*,\text{p-1}}$$

$$E_{n^*,\text{p-1}} = \frac{\left\| \overline{\Psi}^{(n^*+p)} - \overline{\tilde{\Psi}}^{(n^*+p-1)} \right\|_2}{\left\| \overline{\Psi}^{(n^*+1)} - \overline{\tilde{\Psi}}^{(n^*)} \right\|_2}$$

$$C_{p-1}(y) = \text{Chebyshev polynomial of degree } (p-1)$$

$$= \cosh\left[(p-1)\cosh^{-1} y\right], y > 1$$

The polynomials are at least of degree 3 and are terminated when the error reduction factor $E_{n^*,p-1}$ is greater than the theoretical error reduction factor:

$$E_{n^*,\text{p-1}} > \left[C_{p-1}\left(\frac{2-\hat{\sigma}}{\hat{\sigma}}\right)\right]^{-1}$$

The theoretical error reduction factor is the error reduction which would have been achieved if $\hat{\sigma}$ were equal to $\sigma(\overline{\overline{Q}})$, the true dominance ratio. If $E_{n^*,p-1}$ is greater than this, the acceleration cycle has not been as effective as it should have been, so a new cycle is started using the updated dominance ratio estimate, $\hat{\sigma}'$.

Step 3. After the estimates for $\sigma(\overline{\overline{Q}})$ have converged higher degree polynomials are applied.

Step 4. The outer iterations are terminated at outer iteration n if the following three criteria are met:

$$|k^{(n)} - k^{(n-1)}| \le \varepsilon_k$$

$$\frac{\left\| \bar{\Psi}^{(n)} - \bar{\tilde{\Psi}}^{(n-1)} \right\|_2}{\langle \bar{\Psi}^{(n)}, \bar{\tilde{\Psi}}^{(n-1)} \rangle^{1/2}} \le \varepsilon_{\Psi_2}$$

$$\left(\frac{1}{1-\hat{\sigma}}\right) max_i \left| \frac{\Psi_i^{(n)} - \tilde{\Psi}_i^{(n-1)}}{\Psi_i^{(n)}} \right| < \varepsilon_{\Psi_\infty}$$

where ε_k, ε_{Ψ_2}, and $\varepsilon_{\Psi_\infty}$ are input parameters.

It should be noted that a modification to this basic scheme is made in the actual implementation in NESTLE of the Chebyshev polynomial acceleration. Due to various thermal-hydraulic feedback effects, to be discussed later, the coefficient matrices $\bar{\bar{A}}$ and $\bar{\bar{F}}$ in Equation (27) are changed whenever such effects are accounted for in the system. That is, since feedback effects change cross sections and are dependent upon the flux solution, our matrix problem is truly non-linear since $\bar{\bar{A}}$ and $\bar{\bar{F}}$ depend upon the flux solution. Since the non-linearity is weak, one can guess a flux solution, determine the feedback effects and appropriately modify $\bar{\bar{A}}$ and $\bar{\bar{F}}$, and solve for the flux. This updated flux solution can then be used to reinitiate the cycle until both the feedback and flux solutions converge. One way to handle these effects is to update the $\bar{\bar{A}}$ and $\bar{\bar{F}}$ matrices after complete termination of the outer iteration process. This approach has a clear disadvantage in that it requires large computational time to obtain converged solutions for feedbacks and flux. An alternate approach is to update the coefficient matrix for feedback effects during the Chebyshev acceleration process. In doing so, a substantial reduction in computation time can be realized. The latter approach can be justified by observing that the feedback effects are relatively small perturbations to the original system from a reactor physics point of view and

hence, the entire Chebyshev acceleration scheme is not jeopardized. This modified scheme is incorporated in our work in such a manner that the matrices are updated just before a new Chebyshev polynomial acceleration cycle begins. The same approach is taken in regard to updating the NEM corrections to the coupling coefficients. Figure (1) summarizes the overall nested iterative solution strategy used within the NESTLE code. This strategy has been demonstrated to be efficient and robust.

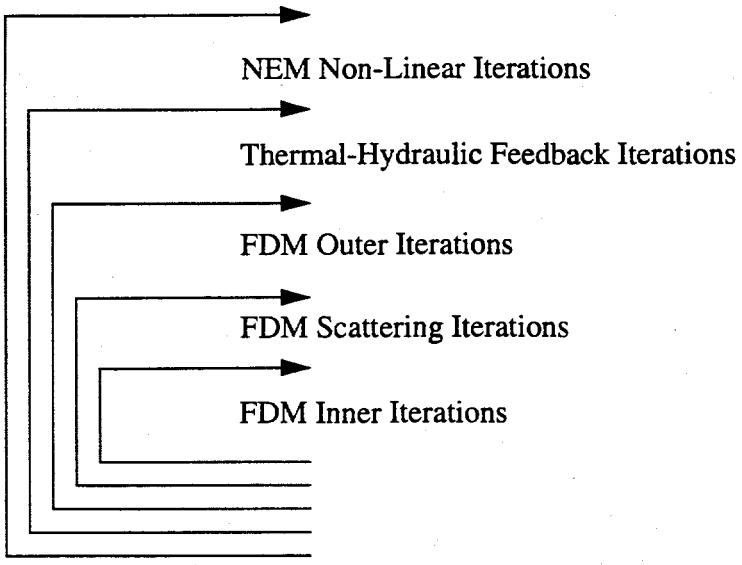


Figure 1: Overview of NESTLE nested iterative solution strategy

II.3 Steady-State Fixed-Source Problem

Real reactors utilize fixed neutron sources to facilitate start-ups and assure high enough count rates for nuclear instrumentation used for control and protection. We refer to the analysis of this situation as a Fixed Source Problem [FSP]. Mathematically, the multi-group diffusion equation for a steady-state FSP is as follows,

$$-\vec{\nabla} \cdot D_g \vec{\nabla} \phi_g + \Sigma_{tg} \phi_g = \sum_{g'=1}^{G} \Sigma_{sgg'} \phi_{g'} + \chi_g \sum_{g'=1}^{G} \nu_{g'} \Sigma_{fg'} \phi_{g'} + S_{ext_g} \tag{40}$$

where $\vec{r}$ dependence has been surpressed and S_{ext_g} denotes the external neutron source

This equation can be solved utilizing nearly exactly the same method as utilized for the AEVP, except now S_{ext_g} appears on the RHS in the NEM equations associated with the AEVP. This applies to both the FDM equation and two-node problem equations. The biggest difference in the solution of the FSP versus AEVP originates because the FSP does not involve determining the fundamental eigenvector. This impacts the outer iterations of the FDM equations in the following manner. For the AEVP, the rate of convergence of the Power Method is determined by the dominance ratio of the outer iterative matrix, $\sigma(\overline{\overline{Q}})$; by contrast, for the FSP the rate of convergence is determined by the spectral radius, $\rho(\overline{\overline{Q}})$, where note that, $\rho(\overline{\overline{Q}}) = k_{eff}$. The implication for the Chebyshev Semi-Iterative method is whenever $\sigma(\overline{\overline{Q}})$ appeared in the governing equations, it should be replaced by $\rho(\overline{\overline{Q}})$. The other implication is that the FSP versus AEVP outer iterations will converge much slower since $\sigma(\overline{\overline{Q}}) < \rho(\overline{\overline{Q}}) = k_{eff} \approx 1$ for problems of interest. A special implementation of the Coarse Mesh Rebalance method, as now described, is utilized for the FSP to accelerate convergence.

II.3.a Fixed-Source Scaling Factor Method

When the FSP is near-critical (*i.e.* $\rho(\bar{\bar{Q}})$ approaches unity), convergence rates even with Chebyshev acceleration are unacceptably slow. This convergence is slow even when the iterative flux shape is correct but the magnitude is in error. To accelerate convergence of the flux magnitude, a global coarse mesh rebalance [12] is proposed. Application of a single scaling prior to the start of a new Chebyshev acceleration cycle sometimes can significantly reduce the required number of outer iterations. This reduction is achieved by an approximate procedure that attempts to scale the current iterative flux vector to the exact flux vector.

For steady-state the FDM based matrix equation analogous to Equation (27) is

$$(\bar{\bar{A}} - \bar{\bar{F}})\,\bar{\phi} = \bar{S}_{ext} \tag{41}$$

Now assume that the q^{th} outer iterative estimate of the flux has the correct shape but is off only in magnitude by a factor of $c^{(q)}$ from the exact solution, *i.e.*

$$\bar{\phi} = c^{(q)}\bar{\phi}^{(q)} \tag{42}$$

Then it follows that an improved q^{th} iterate is given by

$$\hat{\bar{\phi}}^{(q)} = c^{(q)}\bar{\phi}^{(q)} \tag{43}$$

or in terms of the fission source

$$\hat{\bar{\Psi}}^{(q)} = c^{(q)}\tilde{\bar{\Psi}}^{(q)} \tag{44}$$

where $\tilde{\bar{\Psi}}^{(q)}$ is the Chebyshev accelerated fission source. The *fixed source scaling factor*, $c^{(q)}$ is defined so as to preserve neutron balance in an integral sense. Utilizing Galarkin weighting defines $c^{(q)}$ as follows

$$c^{(q)} = \frac{\langle \bar{\phi}^{(q)}, \bar{S}_{ext} \rangle}{\langle \bar{\phi}^{(q)}, (\bar{\bar{A}}(c^{(q)}) - \bar{\bar{F}}(c^{(q)}))\,\bar{\phi}^{(q)} \rangle} \tag{45}$$

This method does differ from the fundamental mode contamination adjustment approach used in DIF3D [10].

The dependence on the scale factor of the matrix operators indicated in Equation (45) originates because of thermal-hydraulic (T-H) feedback, implying the solution of Equation (45) for $c^{(q)}$ involves a non-linear root search. Difficulty in this search originates because the Equation (45) RHS has a singularity, which is addressed as follows. It is known that when the reactor is close to critical, the flux can be approximated by the AEVP flux. This implies that the Equation (45) RHS can be approximated as

$$\cong \left[\frac{\lambda_0(c^{(q)})}{1-\lambda_0(c^{(q)})} \right] \left[\frac{\langle \bar{\phi}^{(q)}, \bar{S}_{ext} \rangle}{\langle \bar{\phi}^{(q)}, \bar{\bar{F}}(c^{(q)})\bar{\phi}^{(q)} \rangle} \right] \tag{46}$$

where λ_0 is the eigenvalue (*i.e.* $\lambda_0 = k_{eff}$). Since the second bracketed term varies much slower than the first bracketed term as $c^{(q)}$ varies for a near critical system, the second term is treated as constant. We next assume that $\lambda_0(c^{(q)})$ varies linearly with $c^{(q)}$.

$$\lambda_0(c^{(q)}) = \lambda_0(c_1^{(q)}) + \frac{\lambda_0(c_2^{(q)}) - \lambda_0(c_1^{(q)})}{(c_2^{(q)} - c_1^{(q)})} (c^{(q)} - c_1^{(q)}) \tag{47}$$

The values of $\lambda_0(c_1^{(q)})$ and $\lambda_0(c_2^{(q)})$ are obtained by explicitly evaluating the Equation (45) RHS for the two scale factor values and using the resulting values in Equation (46) to solve for λ_0 values. Substituting Equation (47) into Equation (46), and using this equation as the RHS of Equation (45) produces a quadratic equation in terms of $c^{(q)}$, with one root denoted $c_3^{(q)}$ being the desired value. For a steady-state problem the following steps are completed to implement the just noted procedure:

Step 1: Calculate 0th outer iterative operator estimates $\bar{\bar{A}}_0$ and $\bar{\bar{F}}_0$, based upon flux $= \bar{\phi}^{(0)}$ flux used in T-H feedback calculations and accounting for external parameters (*e.g.* control rod

position).

Step 2: Solve the FSP iteratively for a fixed number of outer iterations $(\hat{q})$.

Step 3: Set $c_1^{(q)} = 1$ and calculate the following: operator estimates $\bar{\bar{A}}_1$ and $\bar{\bar{F}}_1$ by repeating Step 1 using flux = (Step 2) flux, Equation (45) RHS, and $\lambda_0(c_1^{(q)})$.

Step 4: Set $c_2^{(q)}$ = (Step 3 Equation (45) RHS) and calculate the following: operator estimates $\bar{\bar{A}}_2$ and $\bar{\bar{F}}_2$ by repeating Step 1 using flux $= c_2^{(q)} \times$ Step 2 flux, Equation (45) RHS, and $\lambda_0(c_2^{(q)})$.

Step 5: Solve quadratic equation for $c_3^{(q)}$ and calculate operator estimates $\bar{\bar{A}}_3$ and $\bar{\bar{F}}_3$ by repeating Step 1 using flux $= c_3^{(q)} \times$ Step 2 flux.

This basic process is repeated every so many outer iterations as specified by user input.

The just noted method scales energy groups equally, thus it does not account for the energy spectrum shift that occurs as a result of T-H feedback. This is important in water reactors due to the dependance of moderating power on water density. This effect can be approximately accounted for as follows: Assume that leakage can be approximated by a $D_g B_g^2$ treatment and the Prompt Jump approximation can be used to estimate the flux energy spectrum shift. The values of B_g^2 are spatially dependent and obtained from the current estimate of the flux distribution and prior to the scale factor impact on cross-sections via T-H feedback (*i.e.* after Step 2). Specifically for a two-group problem, suppressing spatial dependance notation, we obtain

$$B_2^2 = \left(\frac{1}{D_2}\right)\left[\Sigma_{r1}\left(\frac{\phi_1^{(q)}}{\phi_2^{(q)}}\right) - \Sigma_{a2}\right] \tag{48}$$

Now an improved estimate for the flux ratio can be obtained as follows, where cross-section

values now reflect the scale factor via T-H feedback (*i.e.* during Steps 3-5).

$$\left(\frac{\phi_1}{\phi_2}\right)_i = \frac{D_2(c_i^{(q)})B_2^2 + \Sigma_{a2}(c_i^{(q)})}{\Sigma_{r1}(c_i^{(q)})} \tag{49}$$

We are now free to set either ϕ_1 or ϕ_2 to the Step 2 flux value, and using Equation (49) solve for the other group flux. NESTLE selects ϕ_1 in its implementation and solves for ϕ_2 . The above method can be generalized to a multi-group formulation and is done so in the NESTLE implementation for the case G=4.

The scaling process can be very effective in obtaining the correct flux magnitude. This avoids a serious problem associated with FSP type problems, which is particularly troubling when the initial guess of the flux is higher than the converged value (*i.e.* approaches from above). However, when the reactor is very close to critical, the scaling process may break down. This is because with high neutron multiplication, $\bar{\bar{A}}\bar{\phi}$ and $\bar{\bar{F}}\bar{\phi}$ are nearly equal and are much larger than $\bar{S}_{ext}$ which implies that to get an accurate estimate of $(\bar{\bar{A}} - \bar{\bar{F}})\bar{\phi}$ a very accurate estimate of the shape of $\bar{\phi}$ is required.

II.3.b Nodal Model - Hexagonal Geometry

II.3.c Eigenvalue Problem

Utilization of NEM for Hexagonal (Hex) geometry introduces several complications not encountered for Cartesian geometry, originating because the surfaces of the Hex do not all align with the Cartesian axis. This can be seen in Figure (2).

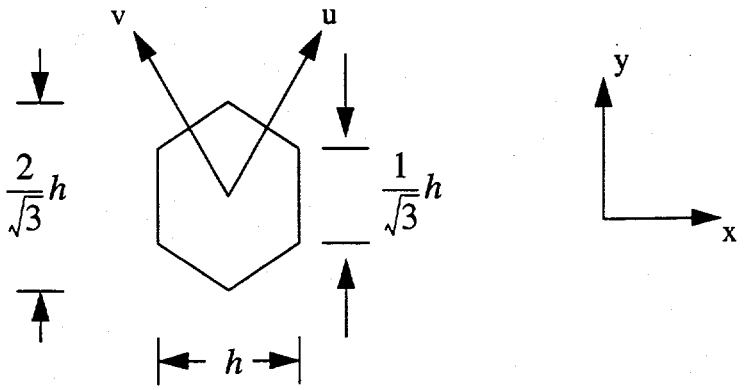


Figure 2: Hex geometry dimensions and axis orientation.

R.D. Lawrence addressed these difficulties in implementing the Hex NEM option in DIF3D [15]. NESTLE utilizes this earlier work, now adapting it for implementation within the context of the non-linear iterative method which facilitates utilization of a higher order transverse leakage treatment.

The derivation of the governing equations for Hex-Z geometry follows the same general approach as for Cartesian geometry. Introducing the transverse directions u and v noted in Figure (2), the nodal balance equation over a Hex is given by

$$(\frac{2}{3h})\,[\bar{L}^l_{g_x}+\bar{L}^l_{g_u}+\bar{L}_{g_v}]+(\frac{1}{\Delta z^l})\,\bar{L}^l_{g_z}+A^l_g\bar{\phi}^l_g=\bar{Q}^l_g \qquad (50)$$

where the $\bar{L}$'s denote as before face-averaged net leakages. Let us first consider the radial plane. By transverse-integration of the diffusion equation over z and y, the one-dimensional balance

equation in direction x is obtained.

$$\frac{dj^l_{g_x}}{dx} + A^l_g \phi^l_{g_x}(x) = Q^l_{g_x}(x) - \frac{2}{\sqrt{3}} j_{g_{xy}}(x,y)\Big|_{-y_s(x)}^{y_s(x)} - \int_{-y_s(x)}^{y_s(x)} dy L^l_{g_z}(x,y) \tag{51}$$

where $\pm y_s(x)$ denote the upper and lower boundaries of the Hex for a given x value; that is,

$$y_s(x) \equiv \frac{1}{\sqrt{3}}(h - |x|) \qquad \text{for } x \in \left[-\frac{h}{2}, \frac{h}{2}\right] \tag{52}$$

$j^l_{g_x}(x)$ denotes the transverse-integrated current in the x direction

$$j^l_{g_x}(x) = \int_{\frac{-\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} dz \int_{-y_s(x)}^{y_s(x)} dy - D^l_g \frac{\partial}{\partial x} \phi^l_g(x,y,z) \tag{53}$$

$j^l_{g_{xy}}(x, \pm y_s(x))$ denotes the z-integrated, surface-normal components of the net current across the u and v directed surfaces

$$j^l_{g_{xy}}(x, \pm y_s(x)) = \mp \int_{\frac{-\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} dz D^l_g \hat{n}_\pm \cdot \vec{\nabla} \phi^l_g(x,y,z)\Big|_{\pm y_s(x)} \tag{54}$$

and $L^l_{g_z}(x,y)$ denotes axial leakage defined by

$$L^l_{g_z}(x,y) = -D^l_g \frac{\partial}{\partial z} \phi^l_g(x,y,z)\Big|_{\frac{-\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} \tag{55}$$

Two additional equations can be defined in a similar manner for the u and v directions. Note that these quantities are neither volume nor surface averaged, which differs from the earlier derivation for Cartesian coordinates. This difference arrises since taking the derivation of the surface-averaged x-directed current appearing in Eqn. (51) would involve derivatives of $y_s(x)$, which introduces algebraic complexity as now discussed.

To solve Eqn. (51) using NEM the one-dimensional surface-averaged flux is expanded in terms of a polynomial expressed as indicated in Eqn. (7) with N=4. The expansion functions $f_n(x)$ for n = 1 and 2 are selected as before, indicated in Eqn. (9). However, due to the behavior of $\phi_{g_x}(x)$ with x, the functions $f_n(x)$ for n = 3 and 4 must be selected differently for Hex. To see this need, evaluate the transverse-integrated current in terms of the transverse-integrated flux, utilizing their definitions and Fick's Law to obtain

$$j^l_{g_x}(x) = -D^l_g \frac{d\phi^l_g}{dx} + D^l_g y'_s(x) \left[\phi^l_{g_{xy}}(x, y_s(x)) + \phi^l_{g_{xy}}(x, -y_s(x))\right] \tag{56}$$

where

$$\phi^l_{g_{xy}}(x, y) = \int_{\frac{-\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} dz \phi^l_g(x, y, z) \tag{57}$$

Now

$$y_s'(x) = -\frac{1}{\sqrt{3}} \operatorname{sgn}(x) \tag{58}$$

is discontinous at x = 0, the node's center-line perpendicular to the x-direction. Since the transverse-integrated current and flux must be continous everywhere, Eqn. (56) implies that the first derivation of the transverse-integrated flux must be discontinous at x = 0; in particular

$$\lim_{\varepsilon \to 0} \left(-D^l_g \frac{d\phi^l_{g_x}}{dx} \right) \Bigg|_{-\varepsilon}^{\varepsilon} = \frac{2D^l_g}{\sqrt{3}} \left[\phi^l_g(x, y_s(x)) + \phi^l_g(x, -y_s(x))\right] \tag{59}$$

To capture this discontinuity and satisfy Eqns. (8) and (10), the functions $f_n(x)$ for n = 3 and 4 are selected as follows.

$$\overbrace{\frac{10}{13}\left(\frac{x}{h}\right)^2 - \frac{1}{2}\left|\frac{x}{h}\right| + \frac{3}{52}}^{f_3} \qquad \overbrace{\left(\frac{x}{h}\right)\left(\left|\frac{x}{h}\right| - \frac{1}{2}\right)}^{f_4} \tag{60}$$

We again have a problem with five unknowns per node and group. Continuity of transverse-integrated current, discontinuity of surface-averaged flux, and the nodal balance equation provide three of the required five equations. The jump discontinuity condition given by Eqn. (59) provides an additional equation, which can be shown to produce [15]

$$a^l_{g_{x3}} = E^l_{g_x}(0) \tag{61}$$

where

$$E^l_{g_x}(x) = \phi^l_g(x, y_s(x)) + \phi^l_g(x, -y_s(x)) - 2\bar{\phi}^l_{g_x}(x) \tag{62}$$

Assuming expressions for $\phi^l_g(x, \pm y_s(s))$ in terms of node average flux and expansion coefficients can be obtained, one then has five unknowns and four equations implying one additional equation is required. This is provided by a Weighted Residual Method, where the weight $\omega_1(x)$ is defined as

$$\omega_1(x) = \text{sgn}(x) \tag{63}$$

Using this weight in a Weighted Residual Method in conjunction with the nodal balance equation can be shown equivalent to preserving half-node nodal balance on each half of the Hex node. The Weighted Residual Method equation that results is

$$\left[A^l_g + 32\frac{D^l_g}{h^2}\right]\bar{\phi}^l_{g_{x1}} = \bar{Q}^l_{g_{x1}} - \frac{2}{3h}\left[\bar{T}^l_{gx} + \bar{T}^l_{gu} - \bar{T}^l_{gv}\right] - \frac{1}{\Delta z^l}\bar{L}^l_{g_{zx1}} + \frac{40}{9}\frac{D^l_g}{h^2}a^l_{g_{x1}} \tag{64}$$

where the following definitions have been introduced

$$\overbrace{\frac{1}{\Delta V^l}\langle \omega_1(x), \phi^l_{g_x}(x)\rangle}^{\bar{\phi}^l_{g_{x1}}} \qquad \overbrace{\frac{1}{\Delta V^l}\langle \omega_1(x), Q^l_{g_x}(x)\rangle}^{\bar{Q}^l_{g_{x1}}} \qquad \overbrace{\frac{\Delta z^l}{\Delta V^l}\langle \omega_1(x), \int_{-y_s(x)}^{y_s(x)} dy L^l_{g_z}(x,y)\rangle}^{\bar{L}^l_{g_{zx1}}} \tag{65}$$

and

$$\bar{T}^l_{g_u} = \frac{\sqrt{3}h}{\Delta V^l}\langle \omega_1(x), j^l_{g_{xy}}(x,y)\,|^{y_s(x)}_{-y_s(x)}\,[\text{u directed component}]\,\rangle$$

$$= \left(\frac{1}{\Delta z^l \cdot \frac{h}{2}}\right)\left[\int_{-h/2}^{0} dx j^l_{g_{xy}}(x,-y_s(x)) + \int_{0}^{h/2} dx j^l_{g_{xy}}(x,y_s(x))\right] \tag{66}$$

$$= \bar{j}^l_{gu}(-h/2) + \bar{j}^l_{gu}(h/2)$$

Substitution of the polynomial expansion for the transverse integrated flux into Eqn. (65) gives $\bar{\phi}^l_{gx1}$ and $\bar{Q}^l_{gx1}$ in terms of the expansion coefficients.

To solve the above equations, we require expressions for $E^l_{gx}(0)$ and $\bar{L}^l_{gzx1}$ in terms of the node average flux and expansion coefficients. As with Cartesian geometry, the transverse leakage in the z-direction will be approximated by a quadratic polynomial. Specifically, following Eqn. (12) one makes the following approximation

$$\frac{2}{\sqrt{3}h}\int_{-y_s(x)}^{y_s(s)} dy L^l_{gz}(x,y) = \bar{L}^l_{gz} + \rho^l_{gz1} f_1(x) + \rho^l_{gz2} f_2(x) \tag{67}$$

where $f_1(x)$ and $f_2(x)$ are defined as previously for Cartesian geometry.

To obtain the expression for $E^l_{gx}(0)$ it can be shown that via a Taylor series expansion about y = 0 that

$$E^l_{gx}(x) = -\frac{1}{6D^l_g}[2y_s(x)]^2 \mathcal{L}^l_{gy}(x) + 0(h^4) \tag{68}$$

where the y-directed leakage is defined as follows.

$$\mathcal{L}^l_{gy}(x) = \frac{1}{2y_s(x)}\left[-D^l_g \frac{\partial}{\partial y}\phi^l_{g_{xy}}(x,y)\right]\Big|^{y_s(x)}_{-y_s(x)} \tag{69}$$

Using the "two-step" approximation (*i.e.* assuming constant transverse leakage over each half-node) produces

$$\mathcal{L}^l_{gy}(x) = \begin{cases} \mathcal{L}^l_{gy-} & \text{for } x \in [-h/2, 0) \\ \mathcal{L}^l_{gy+} & \text{for } x \in (0, h/2] \end{cases} \tag{70}$$

where

$$\mathcal{L}^l_{gy-} = \frac{2}{V^l} \int_{-h/2}^{0} dx - D^l_g \frac{\partial}{\partial y} \phi^l_{gxy}(x, y) \Big|_{-y_s(x)}^{y_s(x)} \tag{71}$$

$$\mathcal{L}^l_{gy+} = \frac{2}{V^l} \int_{0}^{h/2} dx - D^l_g \frac{\partial}{\partial y} \phi^l_{gxy}(x, y) \Big|_{-y_s(x)}^{y_s(x)} \tag{72}$$

Using this approximation and ignoring the $0(h^4)$ term in Eqn. (68) gives

$$E^l_{gx}(0) = \frac{1}{2} [E^l_{gx}(0^-) + E^l_{gx}(0^+)] \tag{73}$$

$$E^l_{gx}(0) = \frac{-h^2}{9D^l_g} [\mathcal{L}^l_{gy-} + \mathcal{L}^l_{gy+}] \tag{74}$$

To complete the evaluation, expressions for $\mathcal{L}^l_{gy-}$ and $\mathcal{L}^l_{gy+}$ in terms of node average flux and expansion coefficients must be determined. This is done via manipulation of previously introduced equations and definitions resulting in the following expression, recalling Eqn. (61)

$$a^l_{gx3} = E^l_{gx}(0) = -\frac{26}{189} \frac{h}{D^l_g} [\bar{L}^l_{gu} + \bar{L}^l_{gv}] - \frac{8}{21} [\bar{\phi}^l_{g_x}(h/2) + \bar{\phi}^l_{g_x}(-h/2) - 2\bar{\phi}^l_g] \tag{75}$$

Since the transverse integrated flux is a function of node average flux and expansion coefficients, Eqn. (75) involves only the unknowns being sought except for the leakage terms.

To complete the formulation of the Hex problem, from Eqn. (56) one recognizes that expressions for $\phi^l_{g_x}(x, \pm y_s(x))$ at $x = \pm h/2$ in terms of the working unknowns is required to evaluate the surface currents. From Eqn. (56) the expression for the face-averaged transverse-integrated current can be obtained

$$\bar{j}^l_{gx} = -D^l_g \frac{d}{dx}\bar{\phi}^l_{gx}(x) + D^l_g \frac{y_s'(x)}{2y_s(x)} E^l_{gx}(x) \tag{76}$$

Again an expression for $E^l_{gx}(x)$ in terms of the working unknowns is required, this time evaluated at $x = \pm h/2$. Using the "two-step" approximation produces

$$E^l_{gx}(\pm h/2) = \frac{1}{18}\frac{1}{D^l_g}\mathcal{L}^l_{gy\pm} \tag{77}$$

and substituting in the expression for $\mathcal{L}^l_{gy\pm}$ gives the following results.

$$\begin{aligned} E^l_{gx}(h/2) = &-\frac{1}{285}\frac{h}{D^l_g}\{17[\bar{j}^l_{gu}(h/2) - \bar{j}^l_{gv}(-h/2)] + 2[\bar{j}^l_{gv}(h/2) - \bar{j}^l_{gu}(-h/2)]\} \\ &-\frac{1}{1235}[179\bar{\phi}^l_{gx}(h/2) + 49\bar{\phi}^l_{gx}(-h/2) - 228\bar{\phi}^l_g] + \frac{1}{130}a^l_{gx3} \end{aligned} \tag{78}$$

and

$$\begin{aligned} E^l_{gx}(-h/2) = &-\frac{1}{285}\frac{h}{D^l_g}\{2[\bar{j}^l_{gu}(h/2) - \bar{j}^l_{gv}(-h/2)] + 17[\bar{j}^l_{gv}(h/2) - \bar{j}^l_{gu}(-h/2)]\} \\ &-\frac{1}{1235}[49\bar{\phi}^l_{gx}(h/2) + 179\bar{\phi}^l_{gx}(-h/2) - 228\bar{\phi}^l_g] + \frac{1}{130}a^l_{gx3} \end{aligned} \tag{79}$$

By combining Eqn. (76) and either Eqn. (78) or (79) we obtain an expression for the surface-averaged transverse current in one direction in terms of currents in the other Hex directions. This does not succeed in eliminating current as an unknown as we desire. This can be addressed as follows:

Since $E^l_{gx}(x)$ at $x = \pm h/2$ are truly continous since the fluxes defining it via Eqn.(62) are continous, and the surface averaged transverse-integrated current is continous everywhere, Eqn.(76) implies that the flux derivative appearing in this equation must be discontinous at $x = \pm h/2$. Employing the above noted properties, the current continuity condition produces the following.

$$\bar{j}^l_{gx}(h/2) = -\frac{D^l_g D^{l+1}_g}{D^l_g + D^{l+1}_g}\left[\left.\frac{d\bar{\phi}^l_g}{dx}\right|_{h/2} + \left.\frac{d\bar{\phi}^{l+1}_g}{dx}\right|_{-h/2}\right] \tag{80}$$

This expression for current is in terms of the expansion coefficients as desired.

Utilizing Eqns. (50), (64), (75), (76) through (79) in the surface-averaged current continuity equation, surface-averaged flux discontinuity equation, and various auxiliary equations relating currents and leakages to flux, we arrive at 13 equations for the 13 unknowns per node per energy groups when considering the x, u, and v directions. Deferring the two-node problem formulation, the z-direction transverse integrated equations will be now developed.

The z-direction transverse integrated equations development follows that for Cartesian geometry except for the transverse leakage terms in the radial plane. The transverse balance equation is given by

$$\frac{d}{dz}\bar{j}_{gz}(z) + A_g^l \bar{\phi}_{gz}^l(z) = \bar{Q}_{gz}^l(z) - \frac{\Delta z^l}{V^l} L_{gxy}^l(z) \tag{81}$$

where the radial plane transverse leakage is defined as

$$L_{gxy}^l(z) = \int_{-h/2}^{h/2} dx \int_{-y_s(x)}^{y_s(x)} dy \left[\frac{\partial}{\partial x} j_{gx}^l(x,y,z) + \frac{\partial}{\partial y} j_{gy}^l(x,y,z) \right] \tag{82}$$

This equation is solved assuming a quartic expansion for the transverse integrated flux as used in Cartesian geometry. The nodal balance (Eqn. (50)), first and second moment Weighted Residual, surface-averaged flux discontinuity, and surface-averaged current continuity equations provide the required number of equations.

The moments of the radial plane transverse leakage that enter the Weighted Residual equations are evaluated utilizing the quadratic approximation to obtain the within node shape; that is

$$\bar{L}_{gxy}^l(z) \cong \bar{L}_{gxy}^l + \rho_{gxy1}^l f_1(z) + \rho_{gxyz}^l f_2(z) \tag{83}$$

where

$$\bar{L}^l_{gxy} = \frac{3h}{2}\frac{1}{V^l}\int_{\frac{-\Delta z^l}{2}}^{\frac{\Delta z^l}{2}} dz L^l_{gxy}(z) \tag{84}$$

and hence

$$\bar{L}^l_{gxy} = \bar{L}^l_{gx} + \bar{L}^l_{gu} + \bar{L}^l_{gv} \tag{85}$$

The expansion coefficients in Eqn. (83) are defined as before [see Eqns. (13)-(15)]. Since the derivation of the Weighted Balance equations is identical to that presented for Cartesian geometry except as noted above in regard to transverse leakages, the interested reader is referred to the earlier presentation.

II.3.d Non-Linear Iterative Strategy

For Hex-Z geometry, the non-linear iterative strategy is applied the same as for Cartesian geometry. For each surface of a node, a two-node problem is solved to obtain the NEM predicted surface-averaged current based upon the FDM flux solution utilizing corrected coupling coefficients. The corrected coupling coefficients are determined demanding that the FDM and NEM predicted currents agree. In the radial plane for Hex geometry, Equation (23) is modified to read as follows.

$$D_{gx+}^{l,FDM} = \frac{2D_g^l D_g^{l+1}}{D_g^l + D_g^{l+1}} \tag{86}$$

Hex Directions (Example - x direction):

Flux Discontinuity

$$d_{gx+}^l \ (a_{gx1}^l + a_{gx2}^l) + d_{gx-}^{l+1} \ (a_{gx1}^{l+1} - a_{gx2}^{l+1}) = 2\,[d_{gx-}^{l+1} \ \phi_g^{l+1} - d_{gx+}^l \ \phi_g^l] \tag{87}$$

Current Continuity

$$\begin{aligned} &-D_g^l\left[(\tfrac{234}{247})\,a_{gx1}^l + (\tfrac{3306}{1235})\,a_{gx2}^l + (\tfrac{18}{65})\,a_{gx3}^l + (\tfrac{1}{2})\,a_{gx4}^l\right] \\ &+D_g^{l+1}\left[(\tfrac{234}{247})\,a_{gx1}^{l+1} - (\tfrac{3306}{1235})\,a_{gx2}^{l+1} - (\tfrac{18}{65})\,a_{gx3}^{l+1} + (\tfrac{1}{2})\,a_{gx4}^{l+1}\right] = \\ &-(\tfrac{h}{285})\,(17\,[\bar{j}_{gv}^{l}(h/2) - \bar{j}_{gu}^{l}(-h/2) + \bar{j}_{gu}^{l+1}(h/2) - \bar{j}_{gv}^{l+1}(-h/2)] \\ &+2\,[\bar{j}_{gu}^{l}(h/2) - \bar{j}_{gv}^{l}(-h/2) + \bar{j}_{gv}^{l+1}(h/2) - \bar{j}_{gu}^{l+1}(-h/2)]\,) \end{aligned} \tag{88}$$

Center Node Jump Condition

$$(\tfrac{8}{21})\,a_{gx2}^l + a_{gx3}^l = -(\tfrac{26}{189})\left(\frac{h}{D_g^l}\right)(\bar{L}_{gu}^l + \bar{L}_{gv}^l) \tag{89}$$

Nodal Balance

$$\begin{aligned} -\left(\frac{D_g^l}{h}\right)(\tfrac{2}{3h})\left[(\tfrac{7068}{1235})\,a_{gx2}^l + (\tfrac{442}{845})\,a_{gx3}^l\right] &= -A_g^l\phi_g^l + \sum_{g'=g}^{G} Q_{gg'}^l \bar{\phi}_{g'}^l \\ -(\tfrac{2}{3h})\,(\tfrac{304}{285})\,(\bar{L}_{gu}^l + \bar{L}_{gv}^l) - \left(\frac{1}{\Delta z^l}\right)\bar{L}_{gz}^l & \end{aligned} \tag{90}$$

Odd Moment Balance

$$\left[(\frac{2}{9})A_g^l+(\frac{240}{171})\left(\frac{D_g^l}{h}\right)(\frac{1}{h})\right]a_{gx1}^l-\left[(\frac{1}{24})A_g^l+2\left(\frac{D_g^l}{h}\right)(\frac{1}{h})\right]a_{gx4}^l$$
$$-\sum_{g'\neq g}^{G}Q_{gg'}^l\left[(\frac{2}{9})a_{g'x1}^l-(\frac{1}{24})a_{g'x4}^l\right]= \tag{91}$$
$$-(\frac{1}{4})\left(\frac{1}{\Delta z^l}\right)\rho_{gz1}^l-(\frac{2}{3h})(\frac{20}{19})[\bar{T}_{gv}^l-\bar{T}_{gu}^l]$$

These equations are supplemented by the following auxiliary equation obtained from Equation(80)

$$\bar{j}_{gx}^l(h/2)=-\left(\frac{D_g^lD_g^{l+1}}{D_g^l+D_g^{l+1}}\right)(\frac{1}{h})\left[(a_{gx1}^l+a_{gx2}^l)+(\frac{36}{13})(a_{gx2}^l-a_{gx2}^{l+1})\right]+ \tag{92}$$
$$(\frac{7}{26})(a_{gx3}^l-a_{gx3}^{l+1})+(\frac{1}{4})(a_{gx4}^l+3a_{gx4}^{l+1})$$

Do note that for the u and v directions, the following mappings of surface currents occur, which impacts the signs of the leakage terms on the RHS of the current continuity and odd moment balance equations:

u direction: $x\rightarrow u,\ v\rightarrow -x,\ u\rightarrow -v$

v direction: $x\rightarrow v,\ u\rightarrow -x,\ v\rightarrow u$

Axial Direction:

Flux Discontinuity

$$d_{gz+}^l\ [a_{gz1}^l+a_{gz2}^l]+d_{gz-}^{l+1}\ [a_{gz1}^{l+1}-a_{gz2}^{l+1}]\ =\ 2\,[d_{gz-}^{l+1}\ \bar{\phi}_g^{l+1}-d_{gz+}^l\ \bar{\phi}_g^l] \tag{93}$$

Current Continuity

$$\frac{D_g^l}{\Delta z^l}\left[a_{gz1}^l+3a_{gz2}^l+\frac{1}{2}a_{gz3}^l+\frac{1}{5}a_{gz4}^l\right]-\frac{D_g^{l+1}}{\Delta z^{l+1}}\left[a_{gz1}^{l+1}-3a_{gz2}^{l+1}+\frac{1}{2}a_{gz3}^{l+1}-\frac{1}{5}a_{gz4}^{l+1}\right]\ =\ 0 \tag{94}$$

Nodal Balance

$$-\left(\frac{D_g^l}{\Delta z^l}\right)\left(\frac{1}{\Delta z^l}\right)\left[6a_{gz2}^l+(\frac{2}{5})a_{gz4}^l\right]\ =\ -(\frac{2}{3h})(\bar{L}_{gx}^l+\bar{L}_{gu}^l+\bar{L}_{gv}^l)-A_g^l\bar{\phi}_g^l+\sum_{g'\neq g}^{G}Q_{gg'}^l\bar{\phi}_{g'}^l \tag{95}$$

First Moment

$$\left[\left(\frac{60}{\Delta z^l}\right)\left(\frac{D_g^l}{\Delta z^l}\right)+A_g^l\right]a_{gz3}^l-\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'z3}^l-10A_g^l a_{gz1}^l+10\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'z1}^l=10\left(\frac{2}{3h}\right)\rho_{gxy1}^l \tag{96}$$

Second Moment

$$\left[\left(\frac{140}{\Delta z^l}\right)\left(\frac{D_g^l}{\Delta z^l}\right)+A_g^l\right]a_{gz4}^l-\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'z4}^l-35A_g^l a_{gz2}^l+35\sum_{g'\neq g}^{G}Q_{gg'}^l a_{g'z2}^l=35\left(\frac{2}{3h}\right)\rho_{gxy2}^l \tag{97}$$

For the z-direction, the same matrix structure as for Cartesian geometry results allowing rearrangement of the associated two-node problem coefficient matrix to achieve reducibility. For the x, u, and v-directions, the two-node problem for two groups can be reduced from a 16 X 16 matrix problem to four 2 X 2 matrix problems and one 8 X 8 matrix problem. For four groups the 32 X 32 matrix problem can be reduced to eight 2 X 2 matrix problems and one 16 X 16 matrix problem. The associated matrix problems are solved analytically to reduce floating point operations required. Having solved the two-node problems, the corrections to the coupling coefficients can be obtained as previously indicated in Equation (22).

II.4 Transient Problem

Under transient conditions, both the multi-group diffusion equation and delayed neutron precursor equations must be solved. These equations, accounting for an external neutron source and utilizing six precursor groups, are given by (suppressing $\vec{r}$ and t dependences for clarity):

$$\frac{1}{v_g}\frac{\partial \phi_g}{\partial t} = \sum_{g'=1}^{G} \Sigma_{s_{gg'}}\phi_{g'} + (1-\beta)\chi_g^{(P)} \sum_{g'=1}^{G} \nu_{g'}\Sigma_{fg'}\phi_{g'} + \sum_{i=1}^{I^{(D)}} \chi_{gi}^{(D)}\lambda_i C_i + \vec{\nabla}\cdot D_g\vec{\nabla}\phi_g - \Sigma_{tg}\phi_g + S_{ext_g} \qquad (98)$$

and

$$\frac{\partial C_i}{\partial t} = \beta_i \sum_{g=1}^{G} \nu_g \Sigma_{fg}\phi_g - \lambda_i C_i \qquad for \qquad i = 1,...,I^{(D)} \qquad (99)$$

where the notation is identical as before except as now noted.

v_g	=	neutron speed for energy group g
$\chi_g^{(P)}$	=	fraction of prompt neutrons born into energy group g
$\chi_{gi}^{(D)}$	=	fraction of delayed neutrons for precursor group i born into energy group g
C_i	=	neutron precursor concentration in precursor group i
λ_i	=	decay constant for precursor group i
β_i	=	fraction of all fission neutrons emitted per fission in precursor group i
β	=	total fraction of fission neutrons which are delayed

Alternately, the 'eigenvalue initiated' transient equations can be obtained from Equations (98) and (99) by setting $S_{ext_g} = 0$ and by replacing $\nu_g\Sigma_{fg}$ with $(\nu_g\Sigma_{fg})/k$ everywhere.

The neutron kinetics equations, Equations (98) and (99), involve differentials in space and time. The time dependence is a difficult problem to treat in neutronics modeling due to the stiffness of the associated equations. The time constants range from very small, associated with prompt neutrons, to very long, associated with the longer lived precursors. NESTLE numerically

treats the temporal dependence in a manner that results in a FSP, which can be solved utilizing the methodology developed for the steady-state FSP.

The first step in this conversion to a FSP is to discretize the time domain into discrete times $\{t_n\}$ and to approximate the time derivative of the flux at time t_{n+1} by a backward difference.

$$\left.\frac{\partial \phi_g}{\partial t}\right|_{t_{n+1}} = \frac{\phi_g(t_{n+1}) - \phi_g(t_n)}{\Delta t_n} \tag{100}$$

This assures unconditional stability. Do note that spatial dependance notation has been and will continue to be suppressed in the equations.

To develop an expression for the precursors' concentrations at time t_{n+1} in terms of the flux $\phi_g(t_{n+1})$, Equation (99) is solved utilizing the Integrating Factor method over time span $[t_n, t_{n+1}]$ to obtain

$$C_i(t_{n+1}) = C_i(t_n)\, e^{-\lambda_i \Delta t_n} + \beta_i \int_{t_n}^{t_{n+1}} \sum_{g=1}^{G} \upsilon_g \Sigma_{fg} \phi_g(t')\, e^{-\lambda_i (t_{n+1} - t')}\, dt' \tag{101}$$

To solve the above integral, a functional form for the time dependent neutron fission source must be assumed over time span $[t_n, t_{n+1}]$. Consistent with the backward difference operator approximation for the time derivative of the flux, the fission source is assumed to vary linearly between time-steps,

$$\nu_g \Sigma_{fg} \phi_g(t) = \nu_g \Sigma_{fg} \phi_g(t_n) + \frac{\nu_g \Sigma_{fg} \phi_g(t_{n+1}) - \nu_g \Sigma_{fg} \phi_g(t_n)}{\Delta t_n} (t - t_n) \tag{102}$$

Incorporating this approximation into Equation (101) and rearranging terms, we obtain

$$C_i(t_{n+1}) = C_i(t_n)\, e^{-\lambda_i \Delta t_n} + F_{i_n}^0 \sum_{g=1}^{G} \nu_g \Sigma_{fg} \phi_g(t_n) + F_{i_n}^1 \sum_{g=1}^{G} \nu_g \Sigma_{fg} \phi_g(t_{n+1}) \tag{103}$$

where

$$F_{i_n}^1 = \frac{\beta_i}{\lambda_i \Delta t_n}\left[\Delta t_n - \frac{1}{\lambda_i}\left(1 - e^{-\lambda_i \Delta t_n}\right)\right] \tag{104}$$

and

$$F_{i_n}^0 = -F_{i_n}^1 + \frac{\beta_i}{\lambda_i}\left(1 - e^{-\lambda_i \Delta t_n}\right) \tag{105}$$

Now substituting Equations (100) and (103) into Equation(98) one obtains

$$\begin{aligned} \frac{1}{\Delta t_n \nu_g}\phi_g(t_{n+1}) - \vec{\nabla}\cdot D_g \vec{\nabla}\phi_g(t_{n+1}) + \Sigma_{t_g}\phi_g(t_{n+1}) &= \sum_{g'=1}^{G} \Sigma_{sgg'}\phi_{g'}(t_{n+1}) \\ + \left\{(1-\beta)\chi_g^{(P)} + \sum_{i=1}^{I^{(D)}} \chi_{gi}^{(D)} \lambda_i F_{i_n}^1\right\} \sum_{g'=1}^{G} \nu_{g'} \Sigma_{fg'} \phi_{g'}(t_{n+1}) &+ S_{eff_g}(t_{n+1}) \end{aligned} \tag{106}$$

where

$$S_{eff_g}(t_{n+1}) = \left[\sum_{i=1}^{I^{(D)}} \chi_{gi}^{(D)} \lambda_i F_{i_n}^0 \sum_{g'=1}^{G} \nu_{g'} \Sigma_{fg'} \phi_{g'}(t_n) + \sum_{i=1}^{I^{(D)}} \chi_{gi}^{(D)} \lambda_i C_i(t_n) e^{-\lambda_i \Delta t_n}\right] + \frac{1}{\Delta t_n \nu_g}\phi_g(t_n) + S_{ext_g}(t_{n+1}) \tag{107}$$

In Equation (106) all cross-sections are evaluated at time t_{n+1}. Inspection of Equation (106) indicates it to be a FSP, with modified operators and source from the steady-state FSP. We refer to Equation (106) as the transient FSP. Hence the application of NEM to the transient FSP and the iterative solution of the resulting coupled equations can proceed exactly the same as for the steady-state FSP. As would be expected, the values of flux and adjusted FDM coupling coefficients at time t_{n+1} for the 0^{th} outer iterative step are based upon their values at time t_n.

If we proceed in this manner, in the two-node problems spatial moments of $S_{eff_g}(t_{n+1})$ would appear. As Equation (107) indicates, $S_{eff_g}(t_{n+1})$ is dependent upon $\{\phi_g(t_n)\}$ and $\{C_i(t_n)\}$. The implication is that the expansion coefficients associated with the transverse integrated fluxes, obtained from solution of the two-node problems, at the previous time t_n must be saved. The same is true for the precursor concentrations, which are treated like the flux for time dependent problems solved by NEM. This would substantially increase the computer memory

requirements.

To overcome this difficulty, further approximations are required in formulating the two-node problems. Recognizing that the within node spatial dependence of $S_{eff_g}(t_{n+1})$ is associated with the contributions from the delay neutrons and the external neutron source, that the external neutron source is assumed to be constant within a node, and that these contributions of neutrons are small, one would expect $S_{eff_g}(t_{n+1})$ within node spatial shape to have little impact on the solution. This justifies treating $S_{eff_g}(t_{n+1})$ spatial dependence approximately. In NESTLE this approximate treatment is done in the same manner as for the transverse leakages; that is, using a quadratic polynomial as indicated in Equation (12). Now only node average values of the flux and precursor concentrations at time t_n must be saved. The node average precursor values are solved for via back substitution using Equation (103) after the node average flux has been computed.

This implementation does create one problem that must be addressed. Since different spatial treatments are used at times t_n and t_{n+1}, the solution of the steady-state FSP and the transient FSP, now for steady-state conditions, will not agree. The practical consequence is that when one utilizes the steady-state FSP solution to determine initial conditions for the transient FSP, the flux will undergo a very mild transient with time even when the initial steady-state conditions are preserved. This annoyance can be avoided by regrouping terms in Equations (106) and (107) as follows.

$$-\vec{\nabla}\cdot D_g\vec{\nabla}\phi_g(t_{n+1})+\Sigma_{tg}\phi_g(t_{n+1}) = \sum_{g'=1}^{G}\Sigma_{sgg'}\phi_{g'}(t_{n+1})+\chi_g\sum_{g'=1}^{G}\nu_{g'}\Sigma_{fg'}\phi_{g'}(t_{n+1})+\tilde{S}_{eff_g}(t_{n+1}) \tag{108}$$

where

$$\chi_g = (1-\beta)\chi_g^{(P)}+\sum_{i=1}^{I^{(D)}}\beta_i\chi_{gi}^{(D)} \tag{109}$$

$$\tilde{S}_{eff_g}(t_{n+1}) = \{\sum_{i=1}^{I^{(D)}} \chi_{gi}^{(D)} \lambda_i F_{i_n}^1 - \beta_i \chi_{gi}^{(D)}\} \sum_{g'=1}^{G} \nu_{g'} \Sigma_{fg'} \phi_{g'}(t_{n+1}) - \frac{1}{\Delta t_n v_g} \phi_g(t_{n+1}) + S_{eff_g}(t_{n+1}) \quad (110)$$

Equation (108) is recognized to be identical to the steady-state FSP except for the replacement of S_{ext_g} with the modified effective source $\tilde{S}_{eff_g}$. Under steady-state conditions, $\tilde{S}_{eff_g}$ equals S_{ext_g} since all the additional terms in Equation (110) cancel. This is true even for the moments of $\tilde{S}_{eff_g}(t_{n+1})$ that appear in the two-node problems provided the within node spatial dependence is treated consistently for the variables appearing in Equation (110) at times t_n and t_{n+1}; in particular, when they are all treated by a quadratic polynomial as previously indicated by Equation (12) for the transverse leakages. The implication is that the transient FSP under steady-state conditions will produce the same solution as produced by the steady-state FSP. This approach has been implemented in NESTLE.

Since the values of node average flux at time t_{n+1}, the unknowns, now appear in the modified effective source, an iterative approach is required. This is easily accomplished within the context of the non-linear iterative method of solving the NEM equations. Recall when solving the two-node problems, node average fluxes are assumed known based upon the latest outer iterative values available for the FDM equation's solution. For transient problems, this corresponds to node averaged flux values at time t_{n+1}, which are precisely the values requires to evaluate the moments of $\tilde{S}_{eff_g}$ that appear in the two-node problems. This approach is utilized within NESTLE to assure the steady-state FSP solution does not "drift" when used as an initial condition in the transient FSP.

II.5 Adjoint Problem

The adjoint solution to the few-group neutron diffusion equation for the eigenvalue problem is of interest. This follows since the adjoint flux can be used to estimate the effect on the reactivity of perturbations via the Raleigh quotient, derived from perturbation theory; and, can be used to estimate kinetics parameters utilizing point-reactor kinetics theory. For these reasons, the ability to solve for the adjoint flux of the eigenvalue problem has been incorporated into the NESTLE code.

For the FDM solution, the development of the equations that need to be solved and their solution are straight forward. The matrix system that needs to be solved for the adjoint flux is obtained by transposing the matrices of the matrix system that is solved for the 'forward' flux. Since the matrices on a per energy group basis are symmetric in space, the only non-symmetric components occur because of energy group coupling originating from scattering and fission. The transpose of the matrices associated with fission and scattering are easily taken and the resulting matrix system solved. Since down-scatter now becomes up-scatter for the transposed scattering operator, one solves the few-group diffusion equation by sweeping from low to high energies (*i.e.* from high energy group number to low energy group number) in the outer iterations. Thermal scattering iterations are completed if up-scatter exists in the 'forward' problem.

For the NEM solution, the situation is more complicated. Mathematically the adjoint solution we seek should not only include the group fluxes but also all the expansion coefficients for the transverse integrated fluxes. This implies that the matrices we should be transposing not only correspond to the nodal balance equations, but also include all the transverse integrated flux constraint equations. These matrices never really appear in the nonlinear NEM iterative method like they would in a more traditional surface current response NEM solution methodology. To overcome this incompatibility and to greatly simplify the adjoint flux solution for the NEM, it is assumed that the diffusion coupling correction coefficients that originate in the nonlinear NEM

iterative method do not change when the core is perturbed. When using the Raleigh quotient to estimate reactivity changes due to core perturbations, this implies that the resulting perturbations to the matrix operators do not include perturbations to the coupling correction coefficients. Since the coupling correction coefficients can be thought of as corrections to the normal FDM diffusion coupling coefficients, this approximation should be acceptable for many applications involving estimating core reactivity.

Having made the above assumption, one need no longer be concerned with evaluating the *adjoint*, transverse integrated flux expansion coefficients, since they only couple to the flux solution through the coupling correction coefficients. Therefore, to obtain the adjoint flux for the NEM one follows exactly the same procedure as for the FDM, except that the coupling correction coefficients determine by the 'forward' solution appear in the matrix operator. Since these coupling correction coefficients make the spatial originated component of the matrices non-symmetric, this must be treated in addition to the non-symmetry in energy groups. This causes no practical problems within NESTLE, where now the transpose of the energy group dependent matrices that appear in the inner iterations are utilized.

A final point needing discussion in regard to solving for the adjoint flux concerns the treatment of cross-section feedback corrections. In NESTLE all feedback effects due to thermal-hydraulics and the transient fission products are frozen at their 'forward' solution determined values. This implies that before an adjoint solution can be completed, a 'forward' solution needs to be completed to obtain the feedback corrected cross-section values. This is done automatically by NESTLE when the adjoint solution option is selected.

To facilitate subsequent utilization of the adjoint flux solution, NESTLE contains the option to write out to a user specified file its values as a function of spatial node and energy group. In this manner other computer codes can utilize this file as input to evaluate, for example, the core's reactivity response to perturbations employing the Raleigh quotient.

II.6 Cross-Section Model

NESTLE has the option of utilizing a macroscopic or microscopic cross-section model. The macroscopic model determines the macroscopic cross-sections used in the diffusion equation solver by characterizing them as a function of the following: node burnup; color (where color refers to the initial composition of the material within the node), rod in or out, coolant density, coolant temperature, effective fuel temperature, and soluble poison concentration. The only isotopic number densities calculated are for the I-Xe and Pm-Sm transient fission products. When used in conjunction with microscopic cross-sections, these isotopes' effects can be directly accounted for. The microscopic cross-section model also determines the isotopic number densities for the U^{234}-U^{236} and U^{238}-Pu^{242} fuel chains, two lumped fission product groups, and a single isotope depletable burnable poison. The relative advantages of both approaches are reduced computational resources and increased accuracy for the macroscopic and microscopic models, respectively. We now discuss the macroscopic model, to be followed by a discussion of the microscopic mode.

II.6.a Macroscopic Model

The macroscopic model represents macroscopic cross-section for a given fuel color, burnup and rod insertion as a Taylor series expansion in terms of coolant density, coolant temperature, effective fuel temperature and soluble poison number density as follows (suppressing fuel color, burnup and rod insertion notation):

$$\hat{\Sigma}_{xg} = a_{1_{xg}} + \sum_{n=1}^{2} a_{(n+1)_{xg}} (\Delta \rho_c)^n + a_{4_{xg}} \Delta T_c + a_{5_{xg}} \Delta \sqrt{T_{F_{eff}}} + \sum_{n=1}^{3} a_{(n+5)_{xg}} (\Delta N_{sp})^n \qquad (111)$$

where

$\hat{\Sigma}_{xg}$ = macroscopic cross-section for reaction type x and energy group g without transient fission products corrected to local conditions

$a_{j_{xg}}$ = expansion coefficients

$\Delta\rho_c = \rho_c - \rho_c^{(0)}$ = change in coolant density (g/cm^3) from reference condition

$\Delta T_C = T_C - T_C^{(0)}$ = change in coolant temperature (°F) from reference condition

$\Delta\sqrt{T_{F_{eff}}} = \sqrt{T_{F_{eff}}} - \sqrt{T_{F_{eff}}^{(0)}}$ = charge in square root of effective fuel temperature (°F) from reference condition

$\Delta N_{sp} = N_{sp} - N_{sp}^{(0)}$ = change in soluble poison number density (cm^{-3} x 10^{-24}) from reference condition.

The soluble poison number density change accounts for both soluble poison concentration (PPM) and coolant density (ρ_c) changes. The effective fuel temperature is evaluated by

$$T_{F_{eff}} = T_C + W_C [W_p T_F + (1 - W_p) T_{F_{Srf}} - T_C] \tag{112}$$

where

W_p = pellet weighting factor, which accounts for resonance flux depression in the interior of the pellet

W_c = core statistical weighting factor, that compensates for the lack of detail in the spatial description of the core

T_F = volume average fuel pellet temperature (°F)

$T_{F_{Srf}}$ = surface average fuel pellet temperature (°F).

To obtain the macroscopic cross-section for node l free of transient fission products, one employs Equation (111) using the expansion coefficients for the fuel color of node l quadratically interpolated to the node l burnup, accounting for control rod effects as follows

$$\hat{\Sigma}_{xg}^{l} = (1 - f_{Rodded}^{l}) (\hat{\Sigma}_{xg}^{l})_{Unrodded} + f_{Rodded}^{l} (\hat{\Sigma}_{xg}^{l})_{Rodded} \tag{113}$$

where f_{Rodded}^{l} is the fraction of node l rodded. This treatment, for coarse axial meshing, produces the artificial behavior of control rod "cusping" when plotting integral rod worth versus insertion depth. Finally correcting for the transient fission products' effect on the absorption cross-section

we obtain

$$\Sigma_{ag}^{l} = \Sigma_{ag}^{l} + \Delta\Sigma_{Xe_{ag}}^{l} + \Delta\Sigma_{Sm_{ag}}^{l} \tag{114}$$

where,

$$\Delta\Sigma_{Xe_{ag}}^{l} = \sigma_{Xe_{ag}}^{l} N_{Xe}^{l}$$
$$\Delta\Sigma_{Sm_{ag}}^{l} = \sigma_{Sm_{ag}}^{l} N_{Sm}^{l}$$

with N_{Xe}^{l} and N_{Sm}^{l} denoting the Xe^{135} and Sm^{135} number densities for node *l*. The Xe and Sm microscopic absorption cross-sections are represented and evaluated in exactly the same manner as the macroscopic cross-sections. As implied by the above equations, the reference conditions for the macroscopic cross-section input are xenon and samarium free.

The following options exist in NESTLE in regard to establishing xenon and samarium number densities: equilibrium, transient, no xenon nor samarium, no xenon and transient samarium, and frozen at restart values. When the option requires, the number densities are determined by solving the I^{135}-Xe^{135} and Pm^{149}-Sm^{149} chain depletion equations. The time-dependent depletion equations for the iodine-xenon chain are given by (again suppressing spatial dependence)

$$\frac{d}{dt}N_{I}^{l}(t) = \gamma_{I}^{l}\sum_{g=1}^{G}\Sigma_{f_g}^{l}(t)\,\phi_{g}^{l}(t) - \lambda_{I}^{l}N_{I}^{l}(t) \tag{115}$$

$$\begin{aligned}\frac{d}{dt}N_{Xe}^{l}(t) &= \lambda_{I}N_{I}^{l}(t) + \gamma_{Xe}^{l}\sum_{g=1}^{G}\Sigma_{f_g}^{l}(t)\,\phi_{g}^{l}(t) - \lambda_{Xe}N_{Xe}^{l}(t) \\ &\quad - \sum_{g=1}^{G}\sigma_{Xe_{ag}}^{l}(t)\,\phi_{g}^{l}(t)\,N_{Xe}^{l}(t)\end{aligned} \tag{116}$$

where subscripts *I* and *Xe* denote I^{135} and Xe^{135}, respectively, and

$N_i^l(t)$ = nuclei number density of isotope i

$\sigma_{i_{ag}}^l(t)$ = microscopic absorption cross section of isotope i

$\Sigma_{f_g}(t)$ = macroscopic fission cross section

$\phi_g^l(t)$ = node average flux

γ_i^l = effective yield (atoms/fission) of isotope i

λ_i = decay constant of isotope i

Likewise, the depletion equations for the Promethium-Samarium chain are given by

$$\frac{d}{dt}N_{Pm}^l(t) = \gamma_{Pm}^l \sum_{g=1}^{G} \Sigma_{f_g}^l(t)\,\phi_g^l(t) - \lambda_{Pm}N_{Pm}^l(t) \tag{117}$$

$$\frac{d}{dt}N_{Sm}^l(t) = \lambda_{Pm}N_{Pm}^l(t) - \sum_{g=1}^{G} \sigma_{Sm_{ag}}^l(t)\,\phi_g^l(t)\,N_{Sm}^l(t) \tag{118}$$

where subscripts *Pm* and *Sm* denote Pm^{149} and Sm^{149}, respectively. The pseudo steady-state solutions of Equations (115) through (118) are given by

$$N_{I_\infty}^l = \frac{\gamma_I^l \sum_{g=1}^{G} \Sigma_{f_g}^l \phi_g^l}{\lambda_I} \tag{119}$$

$$N_{Xe_\infty}^l = \frac{\lambda_I N_{I_\infty}^l + \gamma_{Xe}^l \sum_{g=1}^{G} \Sigma_{f_g}^l \phi_g^l}{\lambda_{Xe} + \sum_{g=1}^{G} \sigma_{Xe_{ag}}^l \phi_g^l} \tag{120}$$

for the Iodine-Xenon depletion chain and,

$$N_{Pm_\infty}^l = \frac{\gamma_{Pm}^l \sum_{g=1}^{G} \Sigma_{fg}^l \phi_g^l}{\lambda_{Pm}} \tag{121}$$

$$N^l_{Sm_-} = \frac{\lambda_{Pm} N^l_{Pm_-}}{\sum_{g=1}^{G} \sigma^l_{Sm_{ag}} \phi^l_g} \tag{122}$$

for the Promethium-Samarium depletion chain. Note that the equilibrium isotopic number densities and flux are coupled. This is addressed by updating the number densities and subsequently cross-sections during the outer iteration process associated with solving the multi-group diffusion equation for either the eigenvalue problem or steady-state FSP.

For the transient solutions, forward differencing over the time-step produces

$$N^l_I(t+\Delta t) = N^l_I(t) + \Delta t\left[\gamma^l_I \sum_{g=1}^{G} \Sigma^l_{f_g}(t)\,\phi^l_g(t) - \lambda_I N^l_I(t)\right] \tag{123}$$

$$\begin{aligned} N^l_{Xe}(t+\Delta t) = {} & N^l_{Xe}(t) + \Delta t\left[\gamma^l_{Xe} \sum_{g=1}^{G} \Sigma^l_{f_g}(t)\,\phi^l_g(t) + \lambda_I N^l_I(t)\right] \\ & -\lambda_{Xe} N^l_{Xe}(t) - \sum_{g=1}^{G} \sigma^l_{Xe_{ag}}(t)\,\phi^l_g(t)\,N^l_{Xe}(t) \end{aligned} \tag{124}$$

$$N^l_{Pm}(t+\Delta t) = N^l_{Pm}(t) + \Delta t\left[\gamma^l_{Pm} \sum_{g=1}^{G} \Sigma^l_{f_g}(t)\,\phi_{g_l}(t) - \lambda_{Pm} N^l_{Pm}(t)\right] \tag{125}$$

$$N^l_{Sm}(t+\Delta t) = N^l_{Sm}(t) + \Delta t\left[\lambda_{Pm} N^l_{Pm}(t) - \sum_{g=1}^{G} \sigma^l_{Sm_{ag}}(t)\,\phi^l_g(t)\,N^l_{Sm}(t)\right] \tag{126}$$

where Δt is the time-step associated with transient fission product conditions. In general, it is selected smaller and larger than the time-steps used in the core depletion and neutron kinetics solutions, respectively.

II.6.b Microscopic Model

The microscopic model only differs from the macroscopic model in that the number densities of the U^{234}-U^{236} and U^{238}-Pu^{242} fuel chains, two lumped fission product groups, and a simple burnable poison are explicitly calculated and used to determine the macroscopic cross-

sections. This implies that the macroscopic cross-section for node l is determined from

$$\hat{\Sigma}^{l}_{x_g} = \hat{\Sigma}^{(Bk)^l}_{x_g} + \sum_{i \in \varsigma_F} \sigma^{l}_{i_{xg}} N^{l}_{i} \tag{127}$$

where

$\hat{\Sigma}^{(Bk)^l}_{x_g}$ = macroscopic cross-section for the background (Bk) isotopes (*i.e.* without isotopes in set ς_F)

ς_F = set of fissile and fertile isotopes, lumped fission products and simple burnable poison (*i.e.* U^{234}-U^{236} and U^{238}-Pu^{242} chains, two lumped fission product groups, and simple burnable poison)

As before, the macroscopic and microscopic cross-sections appearing on the RHS of Equation (127) are determined by interpolating expansion coefficients for the fuel color of node l to the node l burnup and using the results in Equation (111).

The computational effort associated with the microscopic cross-section model is associated with evaluating node dependent microscopic cross-sections and solving the isotopic depletion equations, which can be expressed for the fuel isotopes in general form as follows

$$\frac{d}{dt} N^{l}_{i}(t) = p^{l}_{i}(t)\, N^{l}_{i-1}(t) - d^{l}_{i}(t)\, N^{l}_{i}(t) \tag{128}$$

where

$$p^{l}_{i}(t) = \begin{cases} \lambda_{i-1} \\ \text{or} \\ \sum_{g=1}^{G} \sigma^{l}_{i-1_{cg}}(t)\, \phi^{l}_{g}(t) \end{cases} = \text{production coefficient}$$

$$d_i^l(t) = \lambda_i + \sum_{g=1}^{G} \sigma_{i_{ag}}^l(t)\,\phi_g^l(t) = \text{destruction coefficient}$$

Do note that Plutonium isotopic production is assumed to occur instantaneously upon neutron capture. Equation (128) is solved utilizing the Integrating Factor Technique, assuming the cross-sections constant at the end of time-step values and flux constant, set equal to either the beginning of time-step value (PREDICTOR option) or time-step averaged value (PREDICTOR-CORRECTOR option) producing

$$N_i^l(t_{n+1}) = N_i^l(t_n)\,e^{-d_i^l(t_{n+1})\,(t_{n+1}-t_n)} + p_i^l(t_{n+1}) \int_{t_n}^{t_{n+1}} N_{i-1}^l(t')\,e^{-d_i^l(t_{n+1})\,(t_{n+1}-t')}\,dt' \qquad (129)$$

When using the PREDICTOR-CORRECTOR option, the time-step averaged flux value is given by

$$\langle \phi_g^l \rangle_{n+1} = \frac{1}{2}\left(\phi_g^l(t_n) + \phi_g^l(t_{n+1})\right) \qquad (130)$$

The end of time-step value of the flux appearing in this equation is periodically updated during the flux iterations, resolving the depletion equations for new number densities and subsequently updating cross-sections each time this occurs. The periodicity of updates is specified via code input. For the fissile and fertile chains and simple burnable poison, the integral on the RHS of Equation (129) can be analytically evaluated, since the solutions $\{N_i^l(t)\}$ are composed of linear combinations of exponentials. The analytic solutions of Equation (129) have been determined and used in NESTLE. Ref. [16] provides further details on the analytic solutions.

Two lumped fission products are used to model all fission products except I-Xe and Pm-Sm. Their pseudo number densities are determined by solving the associated pseudo depletion equations. In terms of Equation (128) notation, the lumped fission products production coefficient is given by

$$p^l_{LF_j}(t) = \gamma^l_{LF_j} \sum_{g=1}^{G} \Sigma^l_{fg}(t)\,\phi^l_g(t) \qquad (131)$$

where $\gamma^l_{LF_j}$ is the fission yield for Lumped Fission (LF) product j. Destruction is assumed to not exist.

II.7 Control Option Searches

Various options exist in NESTLE to adjust a certain control parameter such that the core is made critical for the AEVP or achieves the desired specified core power level for the steady-state FSP. Either control rods' position, coolant inlet temperature or soluble poison concentration can be selected as the control parameter to adjust. For the AEVP, core power level can also be selected. NESTLE adjusts the selected control parameter by contrasting the desired eigenvalue (*i.e.* $k_{eff} = 1.0$) or core power level with the current outer iterative predicted value. This data is used to develop a linear expression for the value of the desired core attribute as a function of the selected control parameter, from which a new estimate of the value of the selected control parameter to achieve the desired core attribute value can be estimated. This process is repeated every so many outer iterations until both the convergence criteria on achieving the desired core attribute value and neutronic solution are mutually satisfied. For slowly convergent FSP this approach may fail if the predicted core power level used in developing the linear expression is not adequately converged.

II.8 Hydrodynamic Model

II.8.a Field Equations

The hydrodynamic model used in NESTLE models single and two phase coolant flow up closed coolant channels. A Homogenous Equilibrium Mixture (HEM) model is employed, limiting model applicability to low quality fluids where slip does not occur. The system of equations which describe the average conditions within the flow channel are obtained from the mass continuity and energy conservation equations, assuming pressure to be constant. The constant pressure assumption removes the need to consider the momentum equation. In addition, an Equation of State is used to provide closure.

The one-dimensional, mass continuity equation along a specified channel for a radial node ij is

$$A_C^{ij}(z)\frac{\partial \rho_C^{ij}(z,t)}{\partial t} = -\frac{\partial}{\partial z}G_C^{ij}(z,t)\,A_C^{ij}(z) \tag{132}$$

Similarly, the energy conservation equation assuming constant pressure is given by

$$\begin{aligned} A_C^{ij}(z)\frac{\partial}{\partial t}\left(\rho_C^{ij}(z,t)\,U_C^{ij}(z,t)\right) &= -\frac{\partial}{\partial z}\left(G_C^{ij}(z,t)\,A_C^{ij}(z)\,U_C^{ij}(z,t)\right) \\ -P\frac{\partial}{\partial z}\left(\frac{G_C^{ij}(z,t)\,A_C^{ij}(z)}{\rho_C^{ij}(z,t)}\right) &+ q_S^{ij}(z,t)\,S_F^{ij} + q_C^{ij}(z,t)\,A_C^{ij}(z) \end{aligned} \tag{133}$$

where

ρ_C^{ij} = coolant density

G_C^{ij} = coolant mass velocity

U_C^{ij} = coolant internal energy

q_C^{ij} = volumetric power density from heat deposited directly in the coolant

q_S^{ij} = fuel rod surface heat flux into the coolant

A_C^{ij} = total cross-sectional area for coolant flow within the node

S_F^{ij} = total fuel rod surface area per unit axial length within the node

P = coolant pressure

Note that the field equations contain only the single spatial variable z due to the assumption of a homogenous, closed channel.

The heat flux q_S is obtained using Newton's Law of Cooling as follows

$$q_S^{ij}(z,t) = h_{eff}^{ij}(z,t)\,(T_F^{ij}(z,t) - T_C^{ij}(z,t)) \tag{134}$$

where

T_C^{ij} = coolant temperature

T_F^{ij} = lumped (*i.e.* radially averaged) fuel temperature

h_{eff}^{ij} = effective heat transfer coefficient

The effective heat transfer coefficient, h_{eff} is defined so as to provide the correct heat flux when the lumped versus surface fuel temperature is used in Newton's Law. The coolant temperature T_c is evaluated in terms of coolant internal energy U_c using an Equation of State.

These two partial differential equations contain three unknowns: coolant mass velocity, coolant internal energy and coolant density; a third equation is required to form a closed system of equations. This third equation is provided by an Equation of State expressing coolant density as a function of coolant internal energy. For the steady-state analysis, the field equations are used setting the temporal derivative terms equal to zero.

II.8.b Equation Discretization

In general the field equations are integrated over the flow stream from $z^{k-1/2}$ to $z^{k+1/2}$. This axial spatial mesh defines the volume of a radial node ij, as shown in Figure 3, equivalent to the

neutronic node l. This spatial discretization results in node centered values for the fluid properties ρ_C and U_C, and node boundary values for the mass velocity G_C.

Discretizing the mass continuity equation by integrating over the staggered mesh produces

$$\int_{z^{k-1/2}}^{z^{k+1/2}} A_C^{ij}(z) \frac{\partial \rho_C^{ij}(z,t)}{\partial t} dz = -\int_{z^{k-1/2}}^{z^{k+1/2}} \frac{\partial}{\partial z} (G_C^{ij}(z,t) A_C^{ij}(z)) \, dz \tag{135}$$

Now the Mean Value Theorem is used to approximate the time derivative term to obtain

$$\int_{z^{k-1/2}}^{z^{k+1/2}} A_C^{ij}(z) \frac{\partial \rho_C^{ij}(z,t)}{\partial t} dz = V_C^l \frac{d\bar{\rho}_C^l(t)}{dt} \tag{136}$$

where the *bar* over a variable denotes a node average value. Note that the axial mesh has been selected such that A_c is constant for $z \in [z_{k-1/2}, z_{k+1/2}]$ and $V_C^l = A_C^{ij} \Delta z^k$. The integral of the spatial derivative term yields

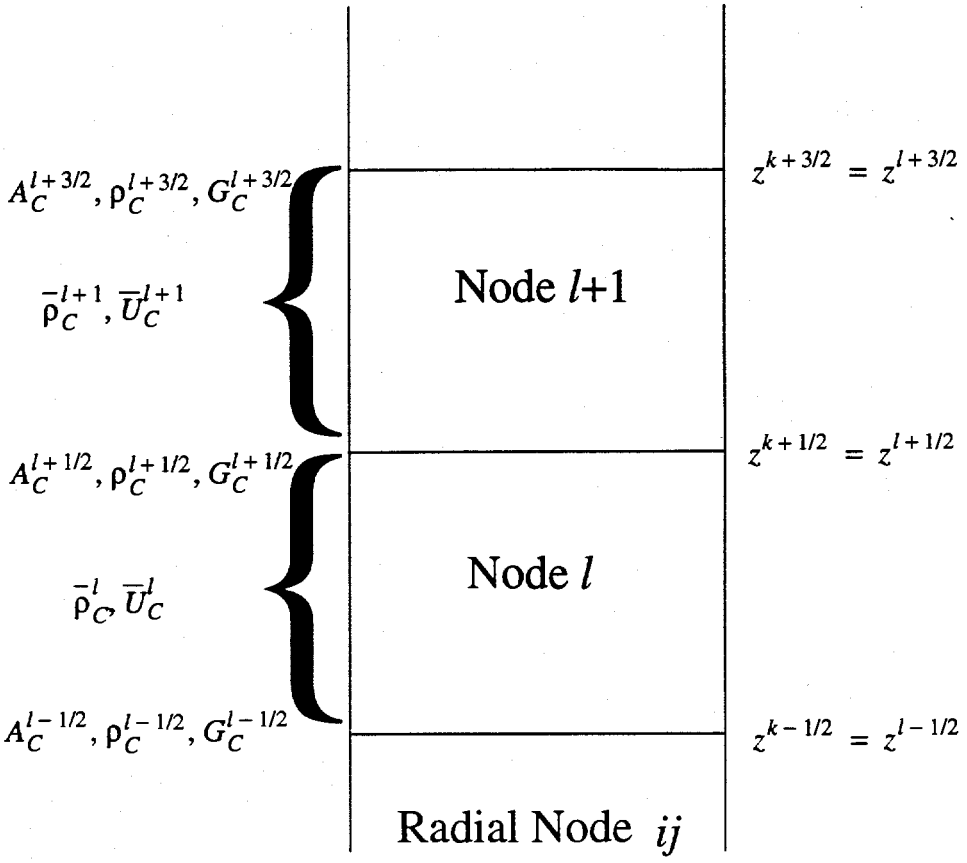


Figure 3: Thermal-hydraulic mesh notation

$$\int_{z^{k-1/2}}^{z^{k+1/2}} \frac{\partial}{\partial z}\left(G_C^{ij}(z,t)\,A_C^{ij}(z)\right)dz = G_C^{l+1/2}(t)\,A_C^{l+1/2} - G_C^{l-1/2}(t)\,A_C^{l-1/2} \tag{137}$$

Finally a backward difference time advancement scheme is employed producing

$$V_C^l \frac{\bar{\rho}^{l,n+1} - \bar{\rho}^{l,n}}{\Delta t_n} + G_C^{l+1/2,n+1} A_C^{l+1/2} - G_C^{l-1/2,n+1} A_C^{l-1/2} = 0 \tag{138}$$

The energy conservation equation, Equation (133) is discretized in a similar manner by integrating along the flow stream to eliminate the spatial derivative, and a semi-implicit time treatment is employed. The convective term is linearized by using new time-step level mass velocity and past time-step level for the other parameters. Furthermore, the coolant density is assumed to be constant over the time-step interval. The result of this discretization scheme is

$$\begin{aligned} V_C^l \bar{\rho}_C^{l,n} \frac{\bar{U}^{l,n+1} - \bar{U}^{l,n}}{\Delta t_n} &= -\left(G_C^{l+1/2,n+1} A_C^{l+1/2} U_C^{l+1/2,n} - G_C^{l-1/2,n+1} A_C^{l-1/2} U_C^{l+1/2,n}\right) \\ &\quad -P\left(\frac{G_C^{l+1/2,n} A_C^{l+1/2}}{\rho_C^{l+1/2,n}} - \frac{G_C^{l-1/2,n} A_C^{l-1/2}}{\rho_C^{l-1/2,n}}\right) + \bar{q}_S^{l,n+1} S_F^l \Delta z^l + \bar{q}_C^{l,n+1} V_F^l \end{aligned} \tag{139}$$

where the time averaged terms appear as a result of the central difference time advancement treatment and are defined as

$$\bar{q}_S^{l,n+1} = \frac{q_S^{l,n} + q_S^{l,n+1}}{2} \tag{140}$$

and

$$\bar{q}_C^{l,n+1} = \frac{q_C^{l,n} + q_C^{l,n+1}}{2} \tag{141}$$

Note the convective and work terms in the energy equation contains fluid property values at the node boundaries. However, as stated previously, the fluid property values are to be calculated as node averages. An intuitive approach would be to spatially average adjacent node average property values along the flow direction to obtain the node boundary values. However, it

has been found that a donor cell averaging technique is necessary for numerical stability [17]. The donor cell average is defined for the coolant internal energy as

$$U_C^{l+1/2,n} = \overline{U}_C^{l,n} \qquad \text{and} \qquad U_C^{l-1/2,n} = \overline{U}_C^{l-1,n} \tag{142}$$

and for the coolant density as

$$\rho_C^{l+1/2,n} = \overline{\rho}_C^{l,n} \qquad \text{and} \qquad \rho_C^{l-1/2,n} = \overline{\rho}_C^{l-1,n} \tag{143}$$

for flow in the positive direction, the only modelling capability within NESTLE.

To solve the two coupled discretized equations, the Equation of State for coolant density in terms of coolant internal energy is linearized as follows

$$\overline{\rho}_C^{l,n+1} \cong \overline{\rho}_C^{l,n} + \frac{\partial}{\partial \overline{U}_C^l}\overline{\rho}_C^l\Big|_{t_n} (\overline{U}_C^{l,n+1} - \overline{U}_C^{l,n}) \tag{144}$$

This equation is then substituted into Equation (138) to solve for the coolant mass velocity in terms of coolant internal energy to produce

$$G_C^{l+1/2,n+1} = \frac{1}{A_C^{l+1/2}}\left[G_C^{l-1/2,n+1} A_C^{l-1/2} - \left(\frac{V_C^l}{\Delta t_n}\right)\left(\frac{\partial}{\partial \overline{U}_C^l}\overline{\rho}_C^l\Big|_{t_n}\right)(\overline{U}_C^{l,n+1} - \overline{U}_C^{l,n}) \right] \tag{145}$$

Finally, the above equation is substituted into Equation (139) to produce an equation only in terms of coolant internal energy

$$\begin{aligned} &\left(\frac{V_C^l}{\Delta t_n}\right)\left(\overline{\rho}_C^{l,n} - \frac{\partial \overline{\rho}_C^l}{\partial \overline{U}_C^l}\Big|_{t_n} A_C^{l+1/2}\overline{U}_C^{l,n}\right)\overline{U}_C^{l,n+1} = G_C^{l-1/2,n+1} A_C^{l-1/2}(\overline{U}_C^{l-1,n} - \overline{U}_C^{l,n}) \\ &-P\left(\frac{G_C^{l+1/2,n} A_C^{l+1/2}}{\overline{\rho}_C^{l,n}} - \frac{G_C^{l-1/2,n} A_C^{l-1/2}}{\overline{\rho}_C^{l-1,n}}\right) + \overline{q}_S^{l,n+1} S_F^l \Delta z^l + \overline{q}_C^{l,n+1} V_C^l + \left(\frac{V_C^l}{\Delta t_n}\right)\overline{\rho}_C^{l,n}\overline{U}_C^{l,n} \end{aligned} \tag{146}$$

This equation is solved for $\overline{U}_C^{l,n+1}$ at each radial node by sweeping in the direction of coolant flow. The following auxiliary relationships are used to evaluate terms appearing in the above equation:

Coolant Volume: $V_C^l = f_{Rodded}^l \cdot V^l$

Total Fuel Rod Surface Area Per Unit Axial Length: $S_F^l = 2\pi r_F N_{FuelPins}^l$

Coolant Volumetric Heat Source: $\bar{q}_C^{l,n} V_C^l = (1 - d_F)\, \bar{q}_T^{l,n} V^l$

where

f_{Rodded}^l = wet fraction for the node when rodded

r_F = fuel rod radius

$N_{FuelPins}^l$ = number of fuel pins within the node

d_F = distribution fraction of energy directly deposited within the fuel

$\bar{q}_T^{l,n}$ = total volumetric power density for heat deposited within the node.

Having obtained values for the coolant internal energy at the new time-step, the coolant densities are evaluated at the new time-step using an Equation of State for the liquid if coolant internal energy indicates sub-cooled conditions for the node. If bulk boiling is indicated, the void fraction is first determined as follows (suppressing superscripts)

$$\bar{\alpha} = \frac{\rho_{C_L}(\bar{U}_C - U_{C_L})}{\rho_{C_V}(U_{C_V} - \bar{U}_C) - \rho_{C_L}(U_{C_L} - \bar{U}_C)} \tag{147}$$

where

U_{C_L} = saturated coolant liquid internal energy

U_{C_V} = saturated coolant vapor internal energy

ρ_{C_L} = saturated coolant liquid density

ρ_{C_V} = saturated coolant vapor density

from which the coolant density is determined as now indicated

$$\bar{\rho}_C = \bar{\alpha}\rho_{C_V} + (1 - \bar{\alpha})\,\rho_{C_L} \tag{148}$$

These same equations are utilized in determining $\frac{d\bar{\rho}_C}{d\bar{U}_C}$ introduced back in Equation (144). Having evaluated the coolant density, Equation (138) is used to solve for the coolant mass velocity at the new time-step.

The above approach is not unconditionally stable and must satisfy the Courant material limit due to the degree of semi-implicitness introduced in linearizing the equations. This stability limit for certain transients restricts the time-step sizes to values smaller than required to control truncation errors. The Stability-Enhancing Two-Step (SETS) Method developed at LANL [18] is used to allow a Courant material limit violating treatment. Since this method was originally utilized within the context of the six-equation model used within the TRAC code, a slight modification of the SETS method is required for the current application.

The stabilizing energy conservation equation used to solve for $\bar{U}_C^{l,n+1}$ is given by

$$V_C^l\left(\frac{\bar{\rho}_C^{l,n+1}\bar{U}_C^{l,n+1}-\bar{\rho}_C^{l,n}\bar{U}_C^{l,n}}{\Delta t_n}\right)=-(G_C^{l+1/2,n+1}A_C^{l+1/2}\bar{U}_C^{l,n+1}$$

$$-G_C^{l-1/2,n+1}A_C^{l-1/2}\bar{U}_C^{l-1,n+1})-P\left(\frac{G_C^{l+1/2,n+1}A_C^{l+1/2}}{\bar{\rho}_C^{l,n+1}}-\frac{G_C^{l-1/2,n+1}A_C^{l-1/2}}{\bar{\rho}_C^{l-1,n+1}}\right) \quad (149)$$

$$+\bar{q}_S^{l,n+1}S_F^l\Delta z^l+\bar{q}_C^{l,n+1}V_C^l$$

Do note the increased degree of implicitness of this equation. Estimates of the current time-step values are determined solving the previous introduced set of equations. Having stabilizing predicted values of coolant internal energy, they are then utilized to update the coolant density as noted before. Also, the coolant temperature is determined based upon coolant internal energy, using an Equation of State for sub-cooled fluid and the saturation temperature for saturated fluid. This approach has been shown to allow large time-steps without stability problems, as indicated by solving the transient equations at steady-state conditions and observing no drift.

The SETS process is repeated as new estimates of the volumetric power density become available, associated with the iterative solution for the flux. To initiate the process, the volumetric

power densities at the new time-step are set equal to the old-time step values. This provides an initial estimate of the coolant and fuel conditions at the new time-step, used to correct cross-sections and commence the flux iterations.

For the feedback correction of cross-sections with respect to coolant density, coolant temperature, and effective fuel temperature, all must be evaluated as node average values. Likewise, the coolant temperature appearing in Newton's Law requires node average values. Previously we stated that we are solving for node average values of coolant properties, so all would seem in order. However, thinking about the steady-state solution of the energy balance equation it becomes clear that coolant internal energy is really evaluated at node boundaries. In this sense donoring is done counter-flow from the node boundary value to the node average value. Therefore the node average coolant temperature is determined using an Equation of State based upon the average of the coolant internal energy at axial elevations *k-1* and *k*. The node average coolant density used to correct cross-sections follows a similar approach, now directly averaging densities. This subtlety becomes important for large axial meshing (*e.g.* 2D radial geometry).

II.8.c Fuel Temperature Model

The lumped (*i.e.* radially averaged) fuel temperature is obtained by utilizing a lumped parameter heat conduction model, in which a simple energy balance for each radial node is performed. This approach should be valid for transients where the fuel pin-wise radial profile of the fuel temperature stays close to the steady-state profile. The rate of energy change in each node, ignoring axial heat conduction, can be expressed as the difference between the node heat source $\bar{q}_F^{ij}(r,z,t)$ and the energy lost due to heat transported radially:

$$\frac{\partial}{\partial t}\left(\rho_F^{ij}(r,z,t)\, U_F^{ij}(r,z,t)\right) = q_F^{ij}(r,z,t) - \vec{\nabla}_r \cdot \vec{q}_{hf}^{ij}(r,z,t) \qquad (150)$$

where r denotes the radial coordinate for an average pin in node *ij* and

U_F^{ij} = fuel internal energy

ρ_F^{ij} = fuel density

q_F^{ij} = volumetric power density from heat deposited directly in the fuel

$\hat{q}_{hf}^{ij}$ = heat flux within the fuel

Using Fourier's Law of Thermal Conductivity expresses the heat flux within the fuel as

$$\hat{q}_{hf}^{ij}(r,z,t) = -k_F^{ij}(r,z,t)\,\vec{\nabla}_r T_F^{ij}(r,z,t) \tag{151}$$

where k_F denotes the fuel's thermal conductivity. Substituting Equation (151) into Equation (150) produces

$$\frac{\partial}{\partial t}\left(\rho_F^{ij}(r,z,t)\,U_F^{ij}(r,z,t)\right) = q_F^{ij}(r,z,t) + \vec{\nabla}_r \cdot k_F^{ij}(r,z,t)\,\vec{\nabla}_r T_F^{ij}(r,z,t) \tag{152}$$

The enthalpy is now expressed in terms of the fuel specific heat (c_{p_F}) and temperature, density is assumed constant, and fuel specific heat is assumed slowly varying in time allowing Equation (152) to be rewritten as

$$\rho_F c_{P_F}^{ij}(r,z,t)\,\frac{\partial T_F^{ij}(r,z,t)}{\partial t} = q_F^{ij}(r,z,t) + \vec{\nabla}_r \cdot k_F^{ij}(r,z,t)\,\vec{\nabla}_r T_F^{ij}(r,z,t) \tag{153}$$

Integrating over the node l volume occupied by fuel, denoted,

$$V_F^l = f_F^l \times V^l \tag{154}$$

where f_F^l indicates the fraction of node l occupied by fuel, applying the central difference time advancement scheme, and rearranging terms yields the final expression for the node average fuel temperature.

$$\begin{aligned}\left[\frac{\rho_F V_F^l}{\Delta t_n} c_{P_F}^{l,n+1} + \left(\frac{h_{eff}^{l,n}}{2}\right) S_F^l \Delta z^l\right] \bar{T}_F^{l,n+1} &= \left(\frac{h_{eff}^{l,n}}{2}\right) S_F^l \Delta z^l \left(\bar{T}_C^{l,n} + \bar{T}_C^{l,n+1}\right) \\ &+ \left[\frac{\rho_F V_F^l}{\Delta t_n} c_{P_F}^{l,n+1} - \left(\frac{h_{eff}^{l,n}}{2}\right) S_F^l \Delta z^l\right] \bar{T}_F^{l,n} + d_F \bar{q}_T^{l,n} V^l\end{aligned} \tag{155}$$

In obtaining Equation (155) the volume integral over the heat conduction term is converted to a surface integral via Green's Theorem, and the resulting expression for the surface heat flux is evaluated using Newton's Law of Cooling given by Equation (134). Do note that the effective heat transfer coefficient is treated explicitly in both the coolant and fuel energy conservation equations. Also the fuel volumetric heat source that appears in Equation (153) has been replaced in Equation (155) using the following expression.

$$\bar{q}_F^{l,n} V_F^l = d_F \bar{q}_T^{l,n} V^l \tag{156}$$

Since V_F^l only appears in the 'heat sink' term in Equation (155) and is a function of f_F^l (see Equation (154)), the value of f_F^l may be varied by fuel color to account for fuel density variations by color to overcome the input limitation of only inputting the core average fuel density. The depletion equations will also correctly reflect fuel density variations captured by the fuel volume fraction.

In addition to lumped fuel temperature, the surface fuel temperature is required to evaluate the effective fuel temperature used to correct cross-sections for Doppler broadening, as indicated in Equation (112). This is obtained by characterizing surface fuel temperature as a function of linear power density for a reference coolant temperature, $T_{C_{Ref}}$ in terms of a polynomial. The spatially dependent linear power density is given by

$$\bar{q}_L^{l,n} = \left(\frac{S_F^l}{N_{FuelPins}^l}\right)\left(\frac{\bar{q}_S^{l,n}}{d_F}\right) \tag{157}$$

The surface fuel temperature is then determined using

$$\bar{T}_{F_{Srf}}^{l,n} = f_{T_{Srf}}(\bar{q}_L^{l,n}) + (\bar{T}_C^{l,n} - T_{C_{Ref}}) \tag{158}$$

where $f_{T_{Srf}}(\bar{q}_L^{l,n})$ denotes the polynomial function.

II.8.d Steady-State Model

For steady-state conditions, the governing equations used to solve for the coolant and fuel conditions are obtained by setting the temporal derivative to zero. When this is done in the coolant's mass continuity equation, Equation (132), it is seen that the product of coolant mass velocity and cross-sectional flow area must be constant up the flow channel. Also the concept of donoring no longer enters since node average coolant values only appeared because of the temporal derivative terms. These implications lead to the following discretized equations:

Coolant Mass Continuity

$$G_C^{l+1/2} = \left(\frac{A_C^{l-1/2}}{A_C^{l+1/2}}\right) G_C^{l-1/2} = \left(\frac{A_{C_{In}}^{ij}}{A_C^{l+1/2}}\right) G_{C_{In}}^{ij} \tag{159}$$

where subscript *In* denotes the inlet to radial node *ij* associated with node *l*.

Coolant Energy Conservation

$$U_C^{l+1/2} = U_C^{l-1/2} - PA_{C_{In}}^{ij} G_{C_{In}}^{ij} \left(\frac{1}{\rho_C^{l+1/2}} - \frac{1}{\rho_C^{l-1/2}}\right) + \bar{q}_C^{l} V_C^{l} + \bar{q}_F^{l} V_F^{l} \tag{160}$$

Fuel Energy Conservation

$$\bar{T}_F^{l} = \bar{T}_C^{l} + \left(\frac{V_F^{l}}{h_{eff}^{l} S_F^{l} \Delta z^{l}}\right) \bar{q}_F^{l} \tag{161}$$

Surface fuel temperature is evaluated as indicated for the transient conditions. These equations are iteratively solved as new estimates of the flux become available, providing new estimates of the surface heat flux and volumetric heat densities. During these iterations the effective heat transfer coefficient is also updated, producing consistent values for the effective heat transfer coefficient and lumped fuel temperature as now described.

II.8.e Effective Heat Transfer Coefficient Evaluation

For the lumped fuel temperature model to be utilized, the effective heat transfer coefficient must be evaluated. For steady-state conditions we can select the effective heat transfer coefficient

such that the correct values of the lumped fuel temperature result, these temperatures determined utilizing a more detailed fuel pellet model. This implies the following:

$$\bar{q}_F A_F = h_{eff} S_F (\bar{T}_F - T_{C_{Ref}}) \tag{162}$$

One can now solve for h_{eff} given values of $\bar{T}_F$ and $\bar{q}_F$ for a fixed coolant temperature as follows.

$$h_{eff} = \frac{\bar{q}_F A_F}{S_F (\bar{T}_F - T_{C_{Ref}})} \tag{163}$$

Note that h_{eff} has been characterized as a function of $\bar{T}_F$ since the fuel thermal conductivity and gap closure, both functions of fuel temperature, are the main reasons why h_{eff} changes. This characterization is captured using a polynomial representation.

For steady-state calculations, an initial estimate of fuel temperature is obtained by characterizing it as a function of linear power density in terms of a polynomial. Given this initial lumped fuel temperature estimate, the effective heat transfer coefficient can be evaluated. Now Equation (161) can be used to calculate a new estimate of the lumped fuel temperature once the node average coolant temperature and volumetric heat density have been evaluated. As the flux solution is iterated, this sequence of calculations is repeated. The iteration of the thermal-hydraulic equations not only addresses feedback between its solution and the neutronic solution, but addresses the non-linearities in calculating the lumped fuel temperature due to effective heat transfer coefficient dependency on fuel temperature.

For transient calculations, the same iterative sequence is employed; however, now the fuel specific heat is also updated due to fuel temperature dependency. In addition the surface heat flux, which appears in the transient coolant energy conservation equation, is also updated utilizing the updated effective heat transfer coefficient, coolant temperature and lumped fuel temperature in Newton's Law.

II.8.f Decay Heat Model

When the reactor shuts down, the reactor power does not immediately drop to zero but

falls off rapidly according to a negative period, eventually determined by the half-life of the longest-lived delayed neutron group. Even then, the transuranics and fission products existing in the fuel continue to decay (β and γ) at decreasing rates for long periods of time. The heat generated from isotopic decay of these isotopes following reactor shutdown is called *decay heat.* Although there are many isotopes involved in the complex decay chain, it is customary to fit a measured decay heat curve for a high burnup reactor with a series of decay heat groups. Thus, the model is analogous to the handling of delayed neutrons.

Accounting for decay heat, the total volumetric heat density is given by

$$q_T(\vec{r},t) = (1-\alpha_T)\sum_{g=1}^{G}\kappa_g\Sigma_{fg}(\vec{r},t)\,\phi_g(\vec{r},t) + \sum_{i=1}^{I^{(DH)}}\varsigma_i D_i(\vec{r},t) \qquad (164)$$

where

$D_i(\vec{r},t)$ = concentration of decay heat group i $\left[\frac{Mev}{cm^3}\right]$

ς_i = disintegration rate (decay constant) $[sec^{-1}]$

α_i = fraction of the total fission energy appearing as decay heat for decay heat group i

α_T = $\sum_{i=1}^{I^{(DH)}}\alpha_i$ = total fraction of the fission energy appearing as decay heat

The concentration of decay heat precursors can be expressed by the following differential equation.

$$\frac{\partial D_i(\vec{r},t)}{\partial t} = \alpha_i\sum_{g=1}^{G}\kappa_g\Sigma_{fg}(\vec{r},t)\,\phi_g(\vec{r},t) - \varsigma_i D_i(\vec{r},t) \qquad \text{for} \qquad i = 1,\ldots,I^{(DH)} \qquad (165)$$

To develop an expression for the decay heat precursor concentrations, a time-integrated expression is derived by integrating Equation (165) from t_n to t_{n+1}. This integration results in

(suppressing $\hat{r}$ dependence for clarity)

$$D_i(t_{n+1}) = D_i(t_n)\, e^{-\varsigma_i \Delta t_n} + \alpha_i \int_{t_n}^{t_{n+1}} \sum_{g=1}^{G} \kappa_g \Sigma_{fg}(t')\, \phi_g(t')\, e^{-\varsigma_i (t_{n+1} - t')}\, dt' \tag{166}$$

To solve the above integral, a functional form for the time dependent neutron fission source density must be developed. Assume the fission source density is constant over the time interval

$t' \in [t_n, t_{n+1}]$ at the past time-step value, *i.e.*

$$\sum_{g=1}^{G} \kappa_g \Sigma_{fg}(t')\, \phi_g(t') = \sum_{g=1}^{G} \kappa_g \Sigma_{fg}(t_n)\, \phi_g(t_n) \tag{167}$$

Incorporating this approximation into Equation (166) and rearranging terms we obtain the desired expression.

$$D_i(t_{n+1}) = D_i(t_n)\, e^{-\varsigma_i \Delta t_n} + \frac{\alpha_i}{\varsigma_i} [1 - e^{-\varsigma_i \Delta t_n}] \sum_{g=1}^{G} \kappa_g \Sigma_{fg}(t_n)\, \phi_g(t_n) \tag{168}$$

In steady-state it is generally assumed that even the longest-lived group is in equilibrium. The steady-state concentration is calculated by setting the time derivative to zero in Equation (165) and solving for the precursor concentration producing,

$$D_{i_\infty} = \frac{\alpha_i}{\varsigma_i} \sum_{g=1}^{G} \kappa_g \Sigma_{fg} \phi_g \tag{169}$$

This equation is utilized to determine the initial conditions required for the transient solution.

III References

[1] B. R. Bandini, *A Three-Dimensional Transient Neutronics Routine for the TRAC-PF1 Reactor Thermal Hydraulic Computer Code*, PhD Dissertation, Pennsylvania State University (1990).

[2] R. D. Lawrence, "Progress in Nodal Methods for the Solution of the Neutron Diffusion and Transport Equations," *Progress in Nuclear Energy*, **17**, No. 3, 271 (1986).

[3] B. A. Finlayson and L. E. Scrivin, "The Method of Weighted Residuals - A Review," *Applied Mechanics Review*, **19**, No. 9, 735 (1966).

[4] K. S. Smith, "Nodal Method Storage Reduction by Non-Linear Iteration," *Trans. Am. Nucl. Soc.*, **44**, 265 (1983).

[5] K. S. Smith, "QPANDA: An Advanced Nodal Method for LWR Analyses," *Trans. Am. Nucl. Soc.*, **50**, 265 (1985).

[6] K. S. Smith and K. R. Rempe, "Testing and Applications of the QPANDA Nodal Method," *Proc. Intl. Topl. Mtg. Advances in Reactor Physics, Mathematics and Computation*, **2**, 861 (1987).

[7] P. R. Engrand, G. I. Maldonado, R. Al-Chalabi and P. J. Turinsky, "Non-Linear Iterative Strategy for NEM: Refinement and Extension," *Trans. Am. Nucl. Soc.*, **65**, 221 (1992).

[8] G. H. Hobson, *Private Communication* (1991).

[9] L. A. Hageman and D. M. Young, *Applied Iterative Methods*, Computer Science and Applied Mathematics, Academic Press, Inc., Orlando (1981).

[10] K. I. Derstine, "DIF3D: A Code to Solve One- Two- and Three-Dimensional Finite-Difference Diffusion Theory Problems," ANL-82-84, Argonne National Laboratory (1984).

[11] R. S. Varga, *Matrix Iterative Analysis*, Prentice Hall, Inc., Englewood Cliffs, New Jersey (1962).

[12] S. Nakamura, *Computational Methods in Engineering and Science*, Wiley and Sons, Inc., New York (1977).

[13] S. K. Zee, *Numerical Algorithms for Parallel Processors Computer Architectures with*

Applications to the Few-Group Neutron Diffusion Equations, PhD thesis, North Carolina State University (1987).

[14] L. A. Hageman and C. J. Pfeifer, "The Utilization of the Neutron Diffusion Program PDQ-5," WAPD-TM-385, Bettis Atomic Power Laboratory, Westinghouse Power Corporation (1965).

[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 (1983).

[16] R. F. Barry, "LEOPARD - A Spectrum Dependent Non-Spatial Depletion Code for the IBM-7094," WCAP-3269-26, Atomic Power Division, Westinghouse Electric Corporation (1963).

[17] T. A. Porsching, J. H. Murphy and J. A. Redfield, "Stable Numerical Integration of Conservation Equations for Hydraulic Networks," *Nuclear Science and Engineering*, **43**, 218 (1971).

[18] J. H. Mahaffy, "A Stability-Enhancing Two-Step Method for Fluid Flow Calculations," *Journal of Computational Physics*, **46**, 239 (1982).

IV User's Guide

The NESTLE code has been written with a view to minimizing the input preparation effort as much as possible. A brief description of the function of each subprogram in NESTLE is given in the Programmer's Guide section along with the flow diagram of the code. By separating the type of input (*e.g.* cross section, geometry, or program control) into distinct input files, it is possible to setup widely varied problems with little input preparation effort. All input data except for restart files are on disk and in free format (except the alphanumeric strings). Thus quick editing is possible and comments to identify each input data can be attached to each data with a double blank between the input and comment. The alphanumeric string variables for file names generally have enough length (A40) so that file names can be assigned to them for later quick identification of the files. Due to the large amount of data written in the restart files, these files are written unformatted to save on storage and facilitate fast retrieval of the data by the code.

The logical units assigned to each file, file name if not free to select, and the contents of the files are as follows. Users specify all "Free to Select" file names in the NESTLE.CNTL file, the exception being Unit 33 files which are specified in the Unit 3 file.

Logical Unit	I/O	File Name	Contents
1	I	NESTLE.CNTL	Code Control Parameter Data
2	I	Free to Select	Geometry Data
3	I	Free to Select	Cross Section Data
4	I	Free to Select	Kinetic Data
5	I	Free to Select	Solution Method Control Data
9	I	Free to Select	Restart Data-Read (Binary)
10	O	Free to Select	Restart Data-Write (Binary)
33	I	Free to Select	Cross Section Data (Optional)
55	O	Free to Select	Hardcopy Output

The following sections describe the input data required for each of the ASCI input files. **If a fixed formatted read is involved,** immediately after the variable name in parenthesis is indicated the applicable format. In addition, for lines of input that are contained within a DO loop or are only read if certain other input values have certain values, a **FORTRAN style DO or IF notation with indentation** is employed.

IV.1 Code Control Parameter Data File

(Unit 1 => "NESTLE.CNTL")

TITLE (A80)

Title line to identify run in hardcopy output

NBUSTEP

Maximum of the number of burnup mask values input for the soluble poison concentration or (burnup steps+1).

NBMAX

Maximum number of control rod banks (groups) to be input.

GEOM (A40)

Geometry data file name (Unit 2)

XSECT (A40)

Cross section data file name (Unit 3)

KINET (A40)

Kinetic data file name (Unit 4)

PERFM (A40)

Solution method control file name (Unit 5)

RESTRT (A40)

Restart file name [to read] (Unit 9)

OUTPUT (A40)

Hardcopy output file name (Unit 55)

OUTADJ (A40)

Unformatted output file name for adjoint solution (Unit 10)

IADJ (A5)

Adjoint option ("Y"/"N")

ISAVEADJ (A5)

Write adjoint solution to file OUTADJ ("Y"/"N")

ITRAN (A5)

Transient option ("Y"/"N")

IRSTRT (A5)

Restart option ("Y"/"N")

ITYPE (A5)

Problem type

Eigenvalue ("EVP")

Fixed-source ("FSP")

NXSEC (A5)

Cross-section corrections to be applied ("Y"/"N")

IF (NXSEC.EQ."Y") THEN READ

ASRCH (A5)

Criticality/Power level search option ("Y"/"N")

IF (ASRCH.EQ."Y") THEN READ

IWHICH

Parameter to search upon

Power level (relative)=1

Soluble poison concentration (PPM)=2

Coolant inlet temperature (^{o}F)=3

Lead (I.D.=highest value) control bank withdrawn (inches)=4

IF(IYPE.EQ."EVP') THEN READ

CKE_TARGET

Target k_{eff} value to match in search

ENDIF EVP

ENDIF ASRCH

NFDBK (A5)

Thermal-hydraulic feedback on ("Y"/"N")

IXE

Xenon-samarium conditions

No Xe and Sm=1

Freeze Xe and Sm=2

Equilibrium Xe and Sm=3

No Xe and transient Sm=4

Transient Xe and Sm=5

ENDIF NXSEC

ABUCKL (A5)

Buckling correction option to be applied ("Y"/"N")

AVEBU(1)

Cycle average burnup at first burnup step (MWD/MTM)

IBURN (A5)

Depletion case ("Y"/"N")

IF IBURN="Y" THEN READ

NBU

Number of burnup steps

(DELBU(I),I=2,NBU+1)

Burnup steps' sizes (MWD/MTM)

ENDIF IBURN

IF (NXSEC.EQ."Y") THEN READ

NPPMX

Number of burnup steps that letdown soluble poison concentration provided at

DO IBU=1,NPPMX

BUPM(IPM),PPM(IPM)

Cycle burnup (MWD/MTM) and soluble poison concentration (PPM)

ENDDO IBU...Burnup step loop

(ZB(IBK),IBK=1,NBMAX)

Control banks (group) axial withdrawal position (in)

PRCNT

Power level (% of full rated)

ENDIF NXSEC

AL3 (A5)

Long input echo option ("Y"/"N")

NPC (A5)

Long hardcopy output option("Y"/"N")

IF (NPC.EQ."Y ')THEN READ

NOUTLONG

Number of variables whose node values are to be output

(AOUTLONG(N),N=1,NOUTLONG) (10(A5,1X))

Names of variables whose node values are to be output

PREL, FLUX, DCOOL, TCOOL, TFUEL, BU, I135, XE135, PM149, SM149, U234,

U235, U236, U238, PU239, PU240, PU241, PU242, LFPG1, LFPG2, BP, ADJFL

ENDIF NPC

CRTON (A5)

CRT output option ("Y"/"N")

DO IBU=1,NBU+1...For each burnup step

ISAVE(IBU) (A5)

Write steady-state restart file ("Y"/"N") at burnup step IBU

IF (ISAVE(IBU).EQ."Y") THEN READ

OUT(IBU) (A40)

Steady-state restart file name [to write] at burnup step IBU

ENDIF ISAVE(IBU)

ENDDO IBU...Burnup step loop

ISAVETR (A5)

Write transient restart file ("Y"/"N")

IF (ISAVETR.EQ."Y") THEN READ

OUTTR (A40)

Transient restart file name [to write]. This file is written at every time indicated by the kinetic file input variable TIMEPR(IT), that indicates times when output should be produced. Note transient restart file is overwritten each time to save space.

ENDIF ISAVETR

IV.2 Geometry Data File

(Unit 2 => "GEOM")

NSHAP (A5)

Node shape in radial plane

Hexagonal ("HEXA")

Cartesian ("CART")

IDRUN (A5)

Core symmetry

IF NSHAP="HEXA"

Full core ("FCORE")

One-third core ("TCORE") {5 to 9 o'clock}

One-sixth core ("SCORE") {5 to 7 o'clock}

One-dimensional axial core ("AXIAL")

IF NSHAP="CART"

Full core ("FCORE")

Half core ("HCORE") {3 to 9 o'clock}

Quarter core ("QCORE") {3 to 6 o'clock}

One-dimensional axial core ("AXIAL")

ENDIF NSHAP

IF (NSHAP.EQ."CART") THEN READ

NX,NY

The x and y total mesh numbers applicable to initial homogenous material regions.

NMULXY

Number of mesh to create from each input x and y material mesh.

ELSEIF (NSHAP.EQ."HEXA") THEN READ

NHR

Number of radial rings of bundles (assemblies) surrounding center bundle.

ENDIF NSHAP

NZ

The z total mesh number applicable to initial material regions.

NMULZ

Number of mesh to create from each input z material mesh.

NFIGURE

Number of different radial configurations (basic figures) of core materials over all elevation.

LIHO,LIPS,LIZU,LIZD

Boundary conditions: Radial exterior, radial interior, z-up, z-down

(reflective=0, zero flux=1, non-reentrant=2, cyclic=3, not applicable=4)

For radial exterior, z-up and z-down boundaries, LIHO **cannot** equal 3.

IF(NSHAP.EQ."CART") THEN READ

BPITCHX,BPITCHY

Pitch for bundles (assemblies) in the x and y directions (in).

NSUBX,NSUBY

Number of different x and y material mesh sizes to utilize in numerical solution.

(NSPACX(I),DDX(I),I=1,NSUBX)

Number of consecutive x material mesh (NSPACX(I)) of constant x mesh size

(DDX(I)) running from west to east. [Note the sum of NSPACX(I) must equal NX.]

(NSPACY(J),DDY(J),J=1,NSUBY)

Number of consecutive y material mesh (NSPACY(J)) of constant y mesh size

(DDY(J)) running from north to south. [Note the sum of NSPACY(J) must equal NY.]

ELSEIF (NSHAP.EQ."HEXA") THEN READ

DELH

Pitch of bundles (assemblies) (in)

ENDIF NSHAP

(See following section entitled GEOMETRY INPUT for detailed description of geometry input.)

DO IB=1,NFIGURE...For each color figure

NBASIC(IB)

Basic figure I.D. (=1,2,...)

DO IY=1,NY...For each y mesh

(NCOL2DT(IX,IY,IB), IX=NXSTART(IY),NXEND(IY),NXSKIP)

Core material colors defined for initial radial material mesh.

ENDDO...Y mesh loop

ENNDO...Color figure loop

Blank Line

DO IY=1,NY...For each y mesh

(NROT2DT(IX,IY),IX=NXSTART(IY),NXEND(IY),NXSKIP)

Mesh clockwise rotation of core material used to define surface ADFs for initial radial material mesh.

(Cartesian: $0^o = 0$, $90^o = 1$, $180^o = 2$, $270^o = 3$)

(Hexagonal: $0^o = 0$, $60^o = 1$, $120^o = 2$, $180^o = 3$, $240^o = 4$, $300^o = 5$)

ENDDO...Y mesh loop

Blank Line

DO IY=1,NY

(NREF2DT(IX,IY),IX=NXSTART(IY),NXEND(IY),NXSKIP)

Mesh diagonal axis reflection of core material used to define surface ADFs for initial radial material mesh. (No reflection = 0,

Reflection about NW to SE node diagonal of original orientation = 1)

ENDDO...Y mesh loop

NSUBZ

Number of different z material mesh sizes to utilize in numerical solution.

(NSPACZ(K),DDZ(K),K=1,NSUBZ)

Number of consecutive z material mesh (NSPACZ(K)) of constant z mesh size (DDZ(K)) running from down to up. [Note the sum of NSPACZ(K) must equal NZ.]

IZCOLS,IZCOLE

Starting and ending axial material mesh numbers for fuel.

(NCOLZT(IZ),IZ=1,NZ)

Basic figure I.D. assigned to initial axial material mesh.

Blank Line

(NTOPZT(IZ),IZ=1,NZ)

Mesh axial reflection of core material used to define surface ADFs for initial axial material mesh. (Up and down surfaces' ADFs reversed)

IF (NXSEC.EQ."Y") THEN READ

Blank Line

DO IY=1,NY...For each y mesh

(LROD2DT(IX,IY),IX=NXSTART(IY),NXEND(IY),NXSKIP)

Control bank (group) I.D. defined for radial material mesh.

(Bank Present=1, 2,...,NBACU, Bank Not Present=0)

ENDDO...Y mesh loop

RODOFFSET

Elevation above bottom of fuel when control bank fully inserted (in).

ENDDO NXSEC

DO IY=1,NY...For each y mesh

(LSEXT2DT(IX,IY),IX=NXSTART(IY),NXEND(IY),NXSKIP)

External source locations defined for radial material mesh

(Source Present=1, Source Not Present=0)

ENDDO...Y mesh loop

Blank Line

(LSEXTZT(IZ),IZ=1,NNZ)

External source locations defined for axial material mesh.

(Source Present=1, Source Not Present=0)

IV.2.a Geometry Input

To simplify the input of geometric information, NESTLE utilizes a highly flexible and yet automated input approach. The required input is all provided on Unit 2 => "GEOM". Core shape (NSHAP) can be either Cartesian-Z or Hexagonal-Z. Core symmetries (IDRUN) for Cartesian-Z include the following: full core, half core (3 to 9 o'clock), quarter core(3 to 6 o'clock), and one-dimensional axial core. Core symmetries (IDRUN) allowed for in Hexagonal-Z include the following: full core, one-third core (5 to 9 o'clock), one-sixth core (5 to 7 o'clock), and one-dimensional axial core. Figure (4) shows examples of radial material geometry figures for each of these geometries. If the number of axial mesh points is set to one and reflective boundary conditions are utilized on z-up and z-down surfaces, a two-dimensional model results.

The number of homogenous material region mesh points input in conjunction with the above core shape and symmetry input is used to automatically determine the input expected for the geometry figures. For Cartesian-Z core shape the number of x (NX) and y (NY) material mesh dictates the Cartesian core layout and hence geometry figures. Figure (4) corresponds to (NX,NY) = (36,36), (36,18) and (18,18) for the full, half and quarter core problems, respectively. For Hexagonal-Z core shape the number of radial rings of bundles surrounding the central bundle (NHR) dictates the hexagonal core layout and hence geometry figures. Figure (4) corresponds to NHR = 10. Note in both these figures the '0' entries in certain material mesh locations. This indicates to the NESTLE code the edge of the geometry to be analyzed where boundary conditions will be applied. **It is required that every material mesh that is created from the core shape, symmetry and material mesh number input have a value input for all geometry figure inputs.**

The material mesh sizes are determined based upon input. For the Hexagonal-Z core shape, the bundle pitch (DELH) uniquely specifies radial mesh size. Variable mesh sizes in the axial direction are allowed. For the Cartesian-Z core shape, variable mesh sizes in all directions

(*i.e.* x, y and z) are allowed. Rather than requiring mesh sizes in each direction to be input for each mesh, mesh sizes are input separately for each direction with the span (NSPACX, NSPACY and NSPACZ) and size (DDX, DDY and DDZ) of a fixed mesh size input over all spans (NSUBX, NSUBY and NSUBZ).

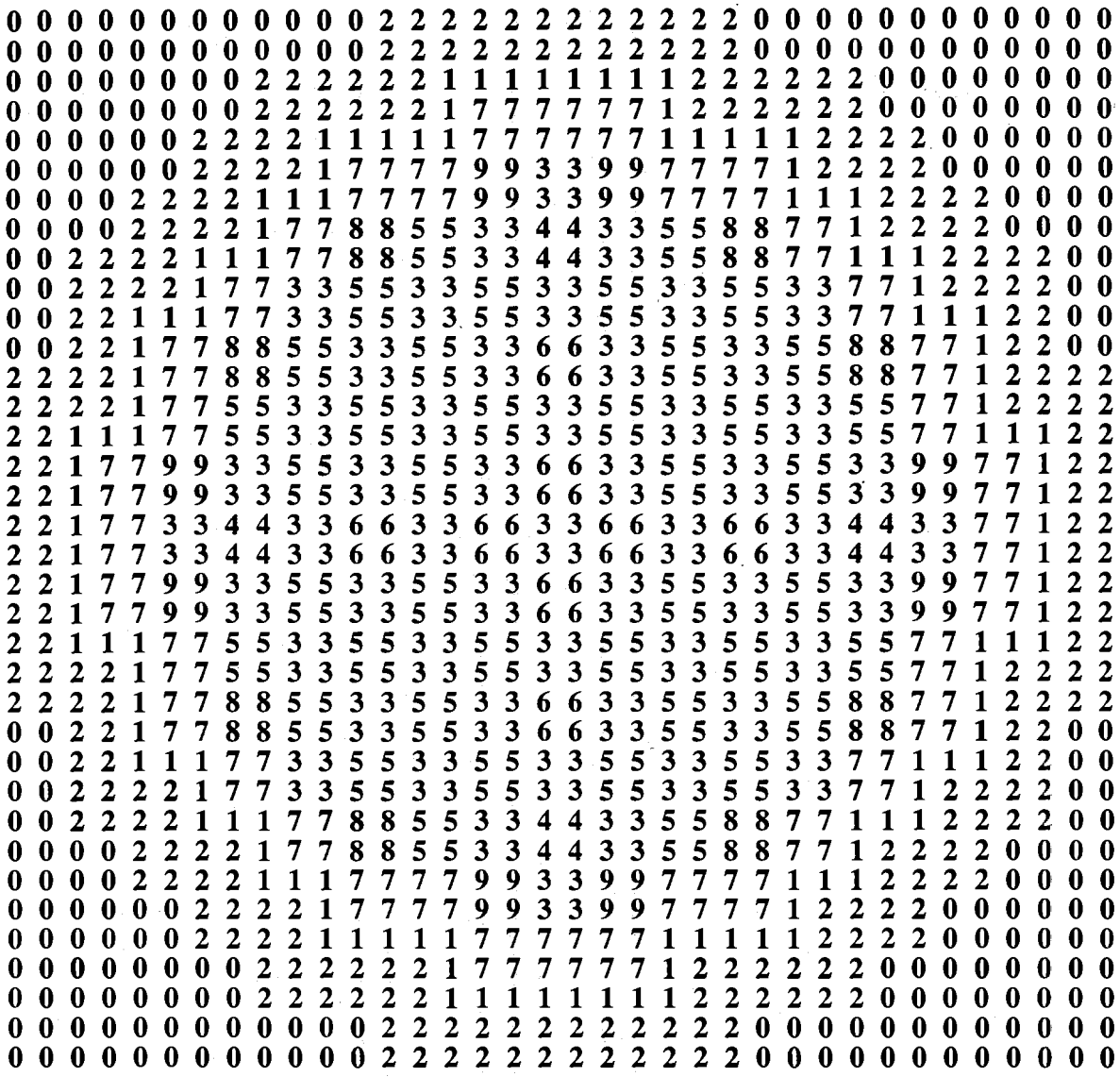
To facilitate mesh refinement **for Cartesian-Z core shape only**, from every initial homogenous material mesh, refined numerical solution meshes can be generated in each direction (*i.e.* x, y and z) each with the same properties as the original material mesh. Feedback and depletion effects will than be applied to the refined numerical solution mesh. This simplifies input and is very convenient when running fine-mesh benchmarks using the FDM. The input variables NMULXY and NMULZ provide the required information to complete the mesh refinement.

Boundary conditions are specified in terms of radial exterior, radial interior, z-up and z-down boundaries. The radial exterior and interior boundaries locations depend upon the core shape and symmetry specified. For example, with Cartesian-Z core shape and quarter core symmetry the interior boundaries correspond to the north and west surfaces and the exterior boundaries correspond to the south and east surfaces. Reflective, zero current, non-reentrant current and cyclic boundary conditions are treated. The cyclic boundary condition may only be used on the radial interior boundaries.

Cross-section (*e.g.* fuel) colors are input via geometry figures for each unique radial configuration (NCOL2DT) as shown in Figure (4). These radial material geometry figures are than assigned to the axial material mesh via an axial mask (NCOLZT). A similar geometry figure input style is used for all input quantities that are spatially dependent.

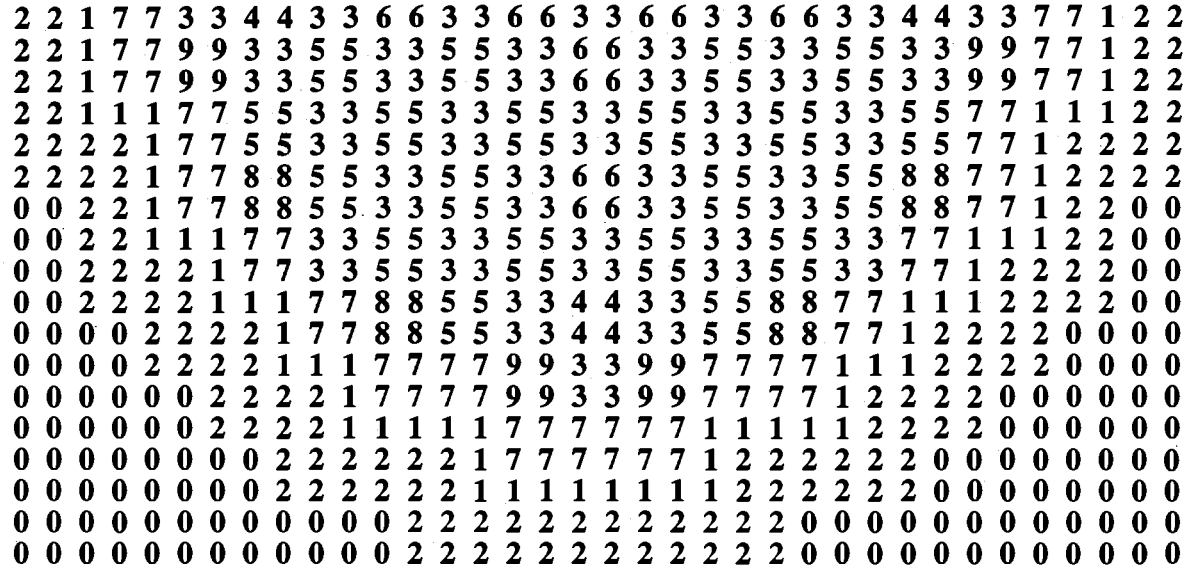
The final point to note with regard to geometry input concerns NESTLE's usage of bundle pitch (*i.e.* BPITCHX and BPITCHY) for Cartesian-Z core shape. These variables are only used to determine the fuel bundle boundaries used in the control of output edits. Given the core shape and symmetry, mesh size and layout, and bundle pitch, NESTLE will automatically determine the

bundle boundaries, including the capability to recognize off-set bundles as encountered in C-E cores and quarter and half assemblies when symmetry does not correspond to full core.

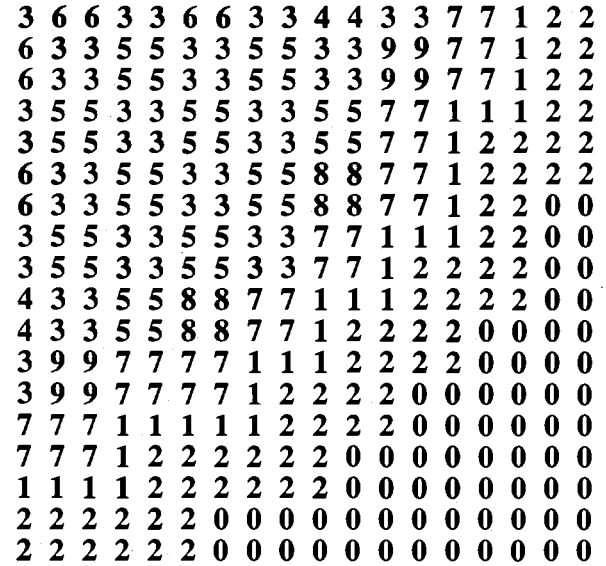


Cartesian-Z Full Core

Figure 4: Radial material geometry figures for different core geometries and symmetries



Cartesian-Z Half Core



Cartesian-Z Quarter Core

Figure 4 (cont): Radial material geometry figures for different core geometries and symmetries

```
          0 0 0 0 5 5 5 0 0 0 0
         0 5 5 5 5 5 5 5 5 5 5 0
        0 5 5 5 5 4 4 4 5 5 5 5 0
       0 5 5 5 4 4 4 4 4 4 5 5 5 0
      5 5 5 4 4 4 1 4 1 4 4 4 5 5 5
     5 5 4 4 4 4 3 3 3 3 4 4 4 4 5 5
    5 5 4 4 1 3 3 3 3 3 3 3 1 4 4 5 5
   0 5 4 4 4 3 3 2 3 3 2 3 3 4 4 4 5 0
  0 5 5 4 1 3 3 3 3 3 3 3 3 3 1 4 5 5 0
 0 5 5 4 4 3 3 3 3 3 3 3 3 3 3 4 4 5 5 0
0 5 5 5 4 4 3 2 3 3 3 3 3 2 3 4 4 5 5 5 0
 0 5 5 4 4 3 3 3 3 3 3 3 3 3 3 4 4 5 5 0
  0 5 5 4 1 3 3 3 3 3 3 3 3 3 1 4 5 5 0
   0 5 4 4 4 3 3 2 3 3 2 3 3 4 4 4 5 0
    5 5 4 4 1 3 3 3 3 3 3 3 1 4 4 5 5
     5 5 4 4 4 4 3 3 3 3 4 4 4 4 5 5
      5 5 5 4 4 4 1 4 1 4 4 4 5 5 5
       0 5 5 5 4 4 4 4 4 4 5 5 5 0
        0 5 5 5 5 4 4 4 5 5 5 5 0
         0 5 5 5 5 5 5 5 5 5 5 0
          0 0 0 0 5 5 5 0 0 0 0
```

Hexagonal-Z Full Core

```
0 5 5 5 4 4 3 2 3 3 3
 0 5 5 4 4 3 3 3 3 3 3
  0 5 5 4 1 3 3 3 3 3 3
   0 5 4 4 4 3 3 2 3 3 2
    5 5 4 4 1 3 3 3 3 3 3
     5 5 4 4 4 4 3 3 3 3 4
      5 5 5 4 4 4 1 4 1 4 4
       0 5 5 5 4 4 4 4 4 4 5
        0 5 5 5 5 4 4 4 5 5 5
         0 5 5 5 5 5 5 5 5 5 5
          0 0 0 0 5 5 5 0 0 0 0
```

Hexagonal-Z Third Core

```
                3
               3 3
              3 3 3
             2 3 3 2
            3 3 3 3 3
           4 3 3 3 3 4
          4 4 1 4 1 4 4
         5 4 4 4 4 4 4 5
        5 5 5 4 4 4 5 5 5
       5 5 5 5 5 5 5 5 5 5
      0 0 0 0 5 5 5 0 0 0 0
```

Hexagonal-Z Sixth Core

Figure 4 (cont): Radial material geometry figures for different core geometries and symmetries

IV.3 Cross-Section Data File

(Unit 3 =>"XSECT" & Unit 33 => *"AXSCIN"*)

NG

Total number of energy groups.

NGT

Number of thermal energy groups.

ICOLXY

Total number of material colors (*i.e.* cross section sets).

NBUMAX

Maximum of the number of burnup mask values input for the cross-sections.

IF(NXSEC.EQ."Y") THEN READ

NPREC

Number of delayed neutron precursor groups.

NDECAY

Number of decay heat precursor groups.

NTERMMACRO,NTERMMACRI

Number of cross-section coefficients input for macroscopic cross-sections w/o and w/ rods in.

NTERMCSCATRO,NTERMCSCATRI

Number of cross-section coefficients input for macroscopic scattering kernels w/o and w/ rods in.

NTERMFPRO,NTERMFPRI

Number of cross-section coefficients input for transient fission products microscopic absorption cross-sections w/o and w/ rods in.

[In following, IXSxxx values imply the following:

Base = 1, Linear-Coolant Density (gm/cm^3) = 2, Quadratic-Coolant Density (gm/cm^3) = 3, Linear-Coolant Temperature (^oF) = 4, Linear in Square Root-Effect. Fuel Temperature (^oF) = 5, Linear-Soluble Poison Number Density (1/b) = 6, Quadratic-Soluble Poison Number Density (1/b) = 7), Cubic-Soluble Poison Number Density (1/b) = 8]

IF(NTERMMACRO.GT.0) READ

& (IXSMACRO(ITERM),ITERM=1,NTERMMACRO)

Cross-section coefficients to be input for macroscopic cross-sections w/o rods in.

IF(NTERMMACRI.GT.0) READ

& (IXSMACRI(ITERM),ITERM=1,NTERMMACRI)

Cross-section coefficients to be input for macroscopic cross-sections w/ rods in.

IF(NTERMCSCATRO.GT.0) READ

& (IXSCSCATRO(ITERM),ITERM=1,NTERMCSCATRO)

Cross-section coefficients to be input for macroscopic scattering kernels w/o rods in.

IF(NTERMCSCATRI.GT.0) READ

& (IXSCSCATRI(ITERM),ITERM=1,NTERMCSCATRI)

Cross-section coefficients to be input for macroscopic scattering kernels w/ rods in.

IF(NTERMFPRO.GT.0) READ

& (IXSFPRO(ITERM),ITERM=1,NTERMFPRO)

Cross-section coefficients to be input for transient fission products microscopic absorption cross-sections w/o rods in.

IF(NTERMFPRI.GT.0) READ

& (IXSFPRI(ITERM),ITERM=1,NTERMFPRI)

Cross-section coefficients to be input for transient fission products microscopic absorption cross-sections w/ rods in.

IMICRO (A5)

Microscopic cross-section option ("Y"/"N").

IF(IMICRO.EQ."Y") THEN READ

NTERMMICRO

Number of cross-section coefficients input for microscopic cross-sections w/o rods in.

NTERMMICRI

Number of cross-section coefficients input for microscopic cross-sections w/ rods in.

[In following, IXSxxx values imply the following:

Base = 1, Linear-Coolant Density (gm/cm^3) = 2, Quadratic-Coolant Density (gm/cm^3) = 3, Linear-Coolant Temperature (^oF) = 4, Linear in Square Root-Effect. Fuel Temperature (^oF) = 5, Linear-Soluble Poison Number Density (1/b) = 6, Quadratic-Soluble Poison Number Density (1/b) = 7), Cubic-Soluble Poison Number Density (1/b) = 8]

IF(NTERMMICRO.GT.0) READ

& (IXSMICRO(ITERM),ITERM=1,NTERMMICRO)

Cross-section coefficients to be input for microscopic cross-sections w/o rods in.

IF(NTERMMICRI.GT.0) READ

& (IXSMICRI(ITERM),ITERM=1,NTERMMICRI)

Cross-section coefficients to be input for microscopic cross-sections w/ rods in.

ENDIF IMICRO

ENDIF NXSEC

AXSEC (A5)

Cross-section colors are read from different input files ("Y"/"N").

IF(NXSEC.EQ."Y") THEN READ

RLI,RLX,PLP

Decay constants (I^{135}, Xe^{135}, Pm^{149}) (1/sec)

(ALAMDA(I),I=1,NPREC)

Delayed neutron precursor decay constants (1/sec).

(XHIDMI(IG,IPREC),IPREC=1,NPREC)

Delayed neutron fission spectrum.

(ZETA(I),I=1,NDECAY)

Decay heat precursors decay rates (1/sec).

IF(IMICRO.EQ."Y") THEN

DO ISOT=1,NISOT... For each isotope (U^{234}, U^{235}, U^{236}, U^{238}, Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242})

(BETAMI(ISOT,IPREC),IPREC=1,NPREC)

Delayed neutron yields for isotope.

ENDDO ISOT...Isotope loop

DO IG=1,NG...For each energy group

(XHIPMI(ISOT,IG),ISOT=1,NISOT)

Prompt neutron spectra for each isotopes (U^{234}, U^{235}, U^{236}, U^{238}, Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242})

ENDDO IG...Energy loop

DO ISOT=1,NISOT... For each isotope (U^{234}, U^{235}, U^{236}, U^{238},

Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242})

(ALPHAI(ISOT,IDECH),IDECH=1,NDECAY)

Decay heat group yields for isotope.

ENDDO ISOT...Isotope loop

DO ISOT=1,NISOT... For each isotope (U^{234}, U^{235}, U^{236}, U^{238},

Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242})

GINMI(ISOT),GXNMI(ISOT),GPNMI(ISOT)

Transient fission product yields (I^{135}, Xe^{135}, Pm^{149}) for isotope.

ENDDO ISOT...Isotope loop

DO IG=1,NG...For each energy group

RNUU4(IG),RNUU5(IG),RNUU6(IG)

Nu values for isotopes (U^{234}, U^{235}, U^{236}).

RNUU8(IG),RNUP9(IG),RNUP0(IG),RNUP1(IG),RNUP2(IG)

Nu values for isotopes (U^{238}, Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242}).

ENDDO IG...Energy group loop

RKU34,RKU35,RKU36

Kappa values (Mev) for isotopes (U^{234}, U^{235}, U^{236}).

RKU38,RKPU39,RKPU40,RKPU41,RKPU42

Kappa values (Mev) for isotopes (U^{238}, Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242}).

GLFP1,GLFP2

Lumped fission products 1 and 2 yields.

ENDIF IMICRO

ENDIF NXSEC

NFUEXY

Number of non-fuel material colors.

Cross section table sets input follow.

Units: Macroscopic cross section (1/cm), microscopic cross sections (barns), burnup (MWD/MTM)

DO ICOL=1,ICOLXY...For each material color

{Note 'I' used below is determined from ICOL and denotes internal storage index of ICOLth material color entry.}

IF(AXSEC.EQ."Y") READ

& AXSCIN (A40)

Name of file containing cross-sections for a specific color.

Following input on file name "XSECT" if AXSEC = "N" and on file name "AXSCIN" if AXSEC = "Y".

Header Line (A80)

NESTLE does not utilize-to help with data file preparation.

Header Line (A80)

NESTLE does not utilize-to help with data file preparation.

AFUEL (A5)

Material cross-sections now to be entered corresponds to a fuel (*i.e.* fissionable) ("Y"/"N").

NCOLOR(I)

Material color number of cross-sections to be entered.

IF(AFUEL.EQ."Y") READ

& NMAX(I)

Number of burnup steps for fuel material color that cross sections provided at. [For non-fuels set equal to 1.]

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

IF(NXSEC.EQ."Y") THEN READ

DO ITERM=1,NTERMMACRI...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XSECRI(N,I,IRX,IG,ITERM),IRX=1,NRXMAX)

Macroscopic cross sections coefficients w/ rods:

IMICRO="Y": transport, absorption.

IMICRO="N": transport, absorption, nu-fission, kappa-fission, nu.

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

DO ITERM=1,NTERMMACRI...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XSECSCRI(N,I,IG,IGP,ITERM),IGP=1,NG-1)

Scattering transfer macroscopic cross sections coefficients w/ rods; from group igp (or igp+1 when upscatter) to group ig. (Note within group (ig to ig) scattering is not entered, hence the non-square matrix.

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

WTFRRO(I),WTFRRI(I),FUFR(I)

Volume fractions for coolant w/o rods, coolant w/ rods, and fuel.

RHOWREF(I),TCOLREF(I),TFREF(I)

Reference conditions cross-sections evaluated at: coolant density (lbm/ft^3), coolant temperature (^oF) and fuel effective temperature (^oF).

(BUBOS(N,I),N=1,NMAC(I))

Burnup values the cross-sections are input at.

DO ITERM=1,NTERMMACRO...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XSECRO(N,I,IRX,IG,ITERM),IRX=1,NRXMAX)

Macroscopic cross sections coefficients w/o rods:

IMICRO="Y": transport, absorption.

IMICRO="N": transport, absorption, nu-fission, kappa-fission, nu.

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

DO ITERM=1,NTERMMACRO...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XSECSCRO(N,I,IG,IGP,ITERM),IGP=1,NG-1)

Scattering transfer macroscopic cross sections coefficients w/o rods; from group igp (or igp+1 when upscatter) to group ig.

(Note within group (ig to ig) scattering is not entered, hence the non-square matrix.

ENDDO ITERM...Cross-section coefficient loop

ENDIF NXSEC

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

B2COL(N,I,IG)

Buckling ($1/cm^2$) {Normally input as 0's for 3D geometries.}

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

IF (AFUEL.EQ."Y") THEN READ

DO N=1,NMAX(I)...For each burnup step

(XHIPN(N,I,IG),IG=1,NG)

Prompt fission neutron spectrum

ENDDO N...Burnup step loop

ENDIF AFUEL

DO N=1,NMAX(I)...For each burnup step.

DO IG=1,NG...For each energy group

(ADFS(I,N,IADF,IG),IADF=1,NADFS)

Assembly Discontinuity Factors w/o rods:

(Cartesian (NADFS=6): N, E, S, W, UP, DOWN)

(Hexagonal (NADFS=8): NE, E, SE, SW, W, NW, UP, DOWN)

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

IF(NXSEC.EQ."Y") THEN READ

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(ADFSRD(I,N,IADF,IG),IADF=1,NADFS)

Assembly Discontinuity Factors w/ rods:

(Cartesian (NADFS=6): N, E, S, W, UP, DOWN)

(Hexagonal (NADFS=8): NE, E, SE, SW, W, NW, UP, DOWN)

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

DO N=1,NMAX(I)...For each burnup step

(VELOCN(N,I,IG),IG=1,NG)

Neutron velocity (cm/sec).

ENDDO N...Burnup step loop

IF (AFUEL.EQ."Y") THEN READ

DO ITERM=1,NTERMFPRO...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XFPNRO(N,I,IG,IFP,ITERM),IFP=1,2)

Microscopic absorption cross-sections coefficients w/o rods: (Sm^{149}, Xe^{135})

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

DO ITERM=1,NTERMFPRI...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(XFPNRI(N,I,IG,IFP,ITERM),IFP=1,2)

Microscopic absorption cross-sections coefficients

w/ rods: (Sm^{149}, Xe^{135})

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

IF(IMICRO.EQ."N") THEN READ

DO N=1,NMAX(I)...For each burnup step

GIN(N,I),GXN(N,I),GPN(N,I)

Transient fission products yields (I^{135}, Xe^{135}, Pm^{149}).

ENDDO N...Burnup step loop

DO N=1,NMAX(I)...For each burnup step

(BETAN(N,I,IPREC),IPREC=1,NPREC)

Delayed neutron yields.

ENDDO N...Burnup step loop

DO N=1,NMAX(I)...For each burnup step

(ALPHAN(N,I,IDECH),IDECH=1,NDECAY)

Decay heat group yields.

ENDDO N...Burnup step loop

ENDIF IMICRO

ENDIF AFUEL

ENDIF NXSEC

IF(NXSEC.EQ."Y") THEN READ

IF (AFUEL.EQ."Y") THEN READ

IF(IMICRO.EQ."Y") THEN READ

Number densities of isotopes in following order (nuclei/cm^3x10^{-24}):

(UPUDEN(I,IRX),IRX=1,3)

(U^{234}, U^{235}, U^{236})

(UPUDEN(I,IRX),IRX=4,8)

U^{238}, Pu^{239}, Pu^{240}, Pu^{241}, Pu^{242})

(UPUDEN(I,IRX),IRX=9,10)

(Lumped FP#1, Lumped FP#2)

UPUDEN(I,11)

(Burnable Poison)

DO ITERM=1,NTERMMICRO...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(UCHAINRO(N,I,IRX,IG,ITERM),IRX=1,6)

(UCHAINRO(N,I,IRX,IG,ITERM),IRX=7,12)

(UCHAINRO(N,I,IRX,IG,ITERM),IRX=13,16)

Microscopic cross sections coefficients w/o rods:

(absorption, fission) for each isotope:

1st line (U^{234}, U^{235}, U^{236})

2nd line (U^{238}, Pu^{239}, Pu^{240})

3rd line (Pu^{241}, Pu^{242})

(UCHAINRO(N,I,IRX,IG,ITERM),IRX=17,18)

Microscopic absorption cross sections coefficients w/o rods: Lumped FP#1, Lumped FP#2

UCHAINRO(N,I,19,IG,ITERM)

Microscopic absorption cross sections coefficients w/o rods: Burnable Poison

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

DO ITERM=1,NTERMMICRI...For each cross-section coefficient

DO N=1,NMAX(I)...For each burnup step

DO IG=1,NG...For each energy group

(UCHAINRI(N,I,IRX,IG,ITERM),IRX=1,6)

(UCHAINRI(N,I,IRX,IG,ITERM),IRX=7,12)

(UCHAINRI(N,I,IRX,IG,ITERM),IRX=13,16)

Microscopic cross sections coefficients w/ rods:

(absorption, fission) for each isotope:

1st line (U^{234}, U^{235}, U^{236})

2nd line (U^{238}, Pu^{239}, Pu^{240})

3rd line (Pu^{241}, Pu^{242})

(UCHAINRI(N,I,IRX,IG,ITERM),IRX=17,18)

Microscopic absorption cross sections coefficients w/ rods: Lumped FP#1, Lumped FP#2

UCHAINRI(N,I,19,IG,ITERM)

Microscopic absorption cross sections coefficients w/ rods: Burnable Poison

ENDDO IG...Energy group loop

ENDDO N...Burnup step loop

ENDDO ITERM...Cross-section coefficient loop

ENDIF IMICRO

ENDIF AFUEL

ENDIF NXSEC

End of input on file name "AXSCIN" if AXSEC = "Y" & returning to file name "XSECT".

ENNDO I...Material color loop

Remaining input on file name "XSECT".

Blank Line

IF (NXSEC.EQ."Y") THEN READ

REFB

Reference soluble poison concentration (PPM)

NFRODS

Number of fuel rods per bundle (assembly)

FUELRAD

Fuel rod radius (in)

WC,WP

Fuel temperature rise and average fuel temperature Doppler effective fuel temperature weight factors.

NPOLTAF

Order of polynomial fitting average fuel temperature versus linear power density.

(COF_TAF(I),I=0,NPOLTAF-1)

Fitting coefficients for average fuel temperature (^{o}F) versus linear power density (kw/ft).

NPOLHFF

Order of polynomial fitting effective heat transfer coefficient versus fuel temperature.

(COF_HFF(I),I=0,NPOLHFF-1)

Fitting coefficients for effective heat transfer coefficient (kw/ft^2-^{o}F) versus fuel temperature (^{o}F).

NPOLTSF

Order of polynomial fitting surface fuel temperature versus linear power density.

(COF_TSF(I),I=0,NPOLTSF-1)

Fitting coefficients for surface fuel temperature (^{o}F) versus

linear power density (kw/ft).

MOLW_COL

Molecular weight of coolant.

ATMW_SOLP

Atomic weight of soluble poison.

ABUN_SOLP

a/o of absorbing isotope of soluble poison (*e.g.* B^{10}).

PIN

Coolant pressure (psia). {Only use within NESTLE is for editing.}

TCOLSAT,UCOLG,RHOG

Coolant saturation temperature (^{o}F), saturated vapor internal energy (BTU/lbm)

and saturated vapor density (lbm/ft^3).

DEPF

Fraction of fission energy deposited directly within fuel.

TCOLIN

Coolant inlet temperature (^{o}F). {Only used for non-transient problems.}

TCOLMIN,TCOLMAX

Coolant inlet temperature (^{o}F) search range for criticality or power level search.

GMASS

Coolant mass velocity (lb/hr-ft^2). {Only used for non-transient problems.}

NPOLUCOL

Order of polynomial fitting coolant liquid internal energy versus temperature.

(COF_UCOL(I),I=0,NPOLUCOL-1)

Fitting coefficients for coolant internal energy (BTU/lbm) versus temperature (^{o}F).

NPOLTCOL

Order of polynomial fitting coolant liquid temperature versus internal energy.

(COF_TCOL(I),I=0,NPOLTCOL-1)

Fitting coefficients for coolant liquid temperature (^{o}F) versus internal energy (BTU/lbm).

NPOLRHOWC

Order of polynomial fitting coolant liquid density versus internal energy.

(COF_RHOWC(I),I=0,NPOLRHOWC-1)

Fitting coefficients for coolant liquid density (lbm/ft^3) versus internal energy (BTU/lbm).

NPOLCPF

Order of polynomial fitting fuel specific heat versus temperature.

(COF_CPF(I),I=0,NPOLCPF-1)

Fitting coefficients for fuel specific heat (BTU/lbm-^{o}F) versus temperature (^{o}F).

RHOWTF

Fuel density (lbm/ft^3).

RATIOHMFUEL

Ratio of heavy metal density to fuel material density.

(Used to convert MTM to MT fuel material)

QQT

Core power density at rated full power (kw/liter).

ENDIF NXSEC

(RT(IG),IG=1,NG)

Initial estimates of relative group flux values.

(SEXT(IG),IG=1,NG)

External source strengths ($1/cm^3$-sec).

IV.4 Kinetic Data File

(Unit 4 =>"KINET")

IF (ITRAN.EQ."Y") THEN READ

NPERT

Number of times that time-dependent input parameters are provided.

NPRINT

Number of times that time-dependent output is to be printed.

NTSPN

Total number of time spans, each utilizing a constant time step size over the time span.

ENDDO ITRAN

IF IXE=4 OR 5 THEN READ

NFP

Number of time-steps for transient fission product case.

ENDIF IXE

IF(ITRAN.EQ."Y")THEN READ

DO IT=1,NPERT...For each transient time perturbation

TIMETR(IT),PMT(IT),TINT(IT),GMINT(IT)

Time (sec), soluble poison concentration (PPM), coolant inlet temperature (^{o}F), coolant mass flow rate (lb/ft^2-sec) (4 values per line)

TIMETR(IT),(ZPT(IBK,IT), IBK=1,NBACU)

Time (sec), control banks steps withdrawn (1+NBACU values per line)

[Note TIMETR(1) must equal 0.]

ENDDO...Transient time perturbation loop

(TIMEPR(IT),IT=1,NPRINT)

Time (sec) output is to be printed.

(TIMESP(IT),DELT(IT),IT=1,NTSPN)

Use time step size DELT(IT) (sec) from time TIMESP(IT-1) to TIMESP(IT) (sec) where TIMESP(0) = 0.

ENDDO ITRAN

IF IXE=4 OR 5 THEN READ

DO IFP=1,NFP...For each fission product transient time perturbation

TIMEFP(IFP),PRFP(IFP),PMFP(IFP),TINFP(IFP)

Time (hrs), core power (relative), soluble poison (PPM), coolant inlet temperature (^{o}F).

[Note that if IBOR="Y" the parameter being searched upon will overwrite the input value of this parameter.]

TIMEFP(IFP),(ZBFP(IBK,IFP),IBK=1,NBACU)

Time (hrs), Control banks steps withdrawn (1+NBACU values per line)

[Note that if IBOR="Y" the parameter being searched upon will overwrite the input value of this parameter.]

ENDDO...Fission product transient time perturbation loop

ENDIF IXE

IV.5 Solution Method Control Data File
(Unit 5 => "PERFM")

N_THRMITR

Maximum number of thermal scattering iterations.

KITR

Maximum number of outer iterations.

EPSK

Outer iteration eigenvalue convergence criteria

EPSOT

Outer iteration L_2 residual convergence criteria

EPSIN

Outer iteration L_{inf} convergence criteria

EPSDET

Inner iteration L_2 relative error reduction convergence criteria

AMETHOD (A5)

Solution method ("FDM"/"NEM")

IF (AMETHOD.EQ "NEM") THEN READ

NNEM

Frequency of NEM coupling coefficients' updates

T-H Off: Number of outer iteration/update

T-H On: Number of T-H calls/update

ENDIF AMETHOD

IF (IBURN.EQ."Y") THEN READ

ADEPL

Depletion method: Predictor or Predictor-Corrector ("PRED"/"CORR")

IF(ADEPL.EQ."CORR") THEN READ

NDELPC

Frequency of Corrector updates for Predictor-Corrector method

T-H Off: Number of outer iteration/update

T-H On: Number of T-H calls/update

ENDIF ADEPL

ENDIF IBURN

ACHEBY (A5)

Chebyshev acceleration on ("Y"/"N")

IF (ACHEBY.EQ."Y" THEN READ

IUPCHE

Upper limit on Chebyshev polynomial order

ENDIF ACHEBY

AISC (A5)

Steady-state scaling factor acceleration on ("Y"/"N")

IF (AISC.EQ."Y") THEN READ

ISC

Outer iteration when steady-state scaling factor acceleration first applied.

ASPSH (A5)

Steady-state spectral shift correction on ("Y"/"N")

ENDIF AISC

IF (ITRAN.EQ."Y") THEN READ

IONE

Number of outer iterations per time step control logic.

Convergence criteria controled=1

Fixed number=2

IF (IONE.EQ.2) THEN READ

IOK

Number of outer iterations per time step.

ENDIF IONE

AISCTR (A5)

Transient scaling factor acceleration on ("Y"/"N")

IF (AISCTR.EQ."Y") THEN READ

ISCTR

Outer iteration when transient scaling factor acceleration first applied.

ASPSHTR

Transient spectral shift correction on ("Y"/"N")

ENDIF AISCTR

ENDIF ITRAN

V Programmer's Guide

This section presents information about the organization and coding of NESTLE that should prove useful to individuals who wish to understand and modify the officially released version of NESTLE. Since NESTLE contains over 23,000 lines of FORTRAN, 95 procedures (*i.e.* subroutines, main program and function subprograms) and 78 fcb files (*i.e.* files containing one or more named COMMON blocks referred to in INCLUDE statements), those wishing to understand the code structure should be prepared to commit significant time to this undertaking.

V.1 Dependence Diagram

The dependence diagram for the NESTLE code is shown in Figure (5). With the exception of the main program, in Figure (5) the subroutines are listed alphabetically indicating the subroutines called one-level deep. Four main control paths exist within the code, associated with the solution of the eigenvalue and steady-state external source, eigenvalue adjoint, transient fission product, and transient neutronics problems. These main control paths utilize many of the same procedures. Where different subroutines are required to perform nearly the same function, a suffix at the end of the subroutine name is utilized to distinguish the function. For example the suffixes 'k' (or 'tr'), 'ss' and 'ad' refer to respectively **k**inetic *(or **tr**ansient)*, steady-state, and **ad**joint; and, the suffixes 'c' and 'h' refer to respectively **C**artesian and **H**exagonal geometries.

V.2 Summary of Procedures

As noted above, there are a total of 95 procedures incorporated into NESTLE. There names and a brief description of their functions are given in Table (2). The header of each procedure contains a brief description via COMMENT lines. Most procedures have very long calling argument lists which are required to allow variables to have problem specific dimensionality without recompilation. This point will be further explained below in the subsection entitle Variables' Storage.

V.3 Variables' Definitions

Rather than define variables throughout the source code of NESTLE, an electronic dictionary has been created. The electronic dictionary consists of the FORTRAN code NESTLE.DICT and the associated data base containing variables' names and definitions. To determine the definition of a variable, the user initiates execution of NESTLE.DICT, indicates that a definition is sought, and enters the variable name. NESTLE.DICT then outputs the variable's definition. This is all done in an interactive manner. Options also exist to add and delete variables' names and definitions. At the conclusion of the NESTLE.DICT execution the data base is updated reflecting any changes made, if so desired. Table (3) shows an interactive session utilizing NESTLE.DICT. It is most conveniently used by opening a separate window where NESTLE.DICT will be controlled from. This code and the associated data base are distributed with the NESTLE code.

V.4 Variables' Storage

Variables with dimensionality are stored either in a container array or in common blocks of fixed dimensionality. The container array (*i.e.* A Array) stores all variables whose dimensions are problem specific, such as number of energy groups and number of spatial nodes. In this manner the dimensionality of variables can be specified at run-time without requiring recompilation for problems with different dimensionality. Also memory is only allocated for variables that will be employed for the specific problem as specified via input. For example, if an eigenvalue problem is to be solved no memory space is allocated for variables that only appear for a neutronic transient problem. The A array is specified (including its size indicated by a PARAMETER statement) in named COMMON block 'array' stored in file 'array.fcb'. The starting address of all variables stored in the A array are determined in SUBROUTINE POINTER. The variable names associated with the starting addresses all have the format 'LXXXXXXN' where 'XXXXXX' is the name of the variable that the starting address is being

given for. The named COMMON block 'varlen' stored in file 'varlen.fcb' contains all the pointer variables.

As noted above, COMMON blocks are utilized to store variables that are either scalar (*i.e.* have no dimensionality) or have problem independent dimensionality. These named COMMON blocks are incorporated into the source code via the 'include' statement using the format 'include yyyyyy.fcb' where 'yyyyyy.fcb' is the file name of a file containing one or more named COMMON blocks. A listing of the fcb files is presented in Table (4). The advantage of the 'include' statement is that modifications to a COMMON block need only be made in one place, the appropriate fcb file. The danger when using the UNIX 'make' feature is that not all procedures which 'include' the altered fcb file will be recompiled since they have not been 'touched'. A sound programming practice when altering fcb files is to 'grep' on *.f files the fcb file names altered and 'touch' all the procedures containing the altered fcb files.

V.5 Machine Specific Instructions

The only machine specific instruction that is utilized by NESTE is the clock call used to provide timing information that is output. Rather than have the machine specific clock instruction scattered throughout the source code, a function subroutine named 'FUNCTION GTIME' is called. Within this subroutine are the machine specific clock calls. Currently the clock call instructions for DEC ULTRIX, IBM AIX and SUN OS operating systems are included, with the clock calls not utilized commented out. In this manner NESTLE can be ported to different platforms and only the file gtime.f needs to be modified.

V.6 Geometry Treatment

Since both Cartesian-Z and Hexagonal-Z geometry can be treated by NESTLE, the geometry treatment needs to be fairly flexible. Most of the setup of the geometry is completed in SUBROUTINE GEOMETRY. For most variables the indices associated with the two radial

directions (IX,IY) are rolled into a single index (IXY) indirectly indexed via IXY=NDNUM(IX,IY). This greatly facilitates handling both Cartesian-Z and Hexagonal-Z geometries. In addition, no memory is wasted to store variables outside the solution domain when a 'raggedy edge' solution domain is employed. In addition to the nodes associated with the solution domain, an extra layer of nodes is added radially about the core such that every surface associated with either an interior boundary (*i.e.* symmetry boundary) or exterior boundary has an extra node located across the surface. This is done to facilitate the various boundary condition treatments. Since these extra nodes IXY index starts after the solution domain nodes IXY index, they can be easily recognized by the value of IXY. In this manner whether a neighbor node is within the solution domain can be determined. If it is determined that the neighbor node is outside the solution domain, than a vector NBC(IXY) contains the information required to apply the specified boundary condition. For cyclic boundary conditions, NBC(IXY) stores the negative value of the IXY of the neighboring node associated with cyclic symmetry. The treatment just noted greatly simplifies code logic in regard to geometry.

The SUBROUTINE GEOMETRY also determines for Cartesian geometry the boundaries of all fuel bundles (assemblies) by using the mesh layout and the bundle pitches in the x and y directions. The capability to treat 'offset' bundles as in C-E cores has been incorporated. The resulting information is utilized to determine bundle average attributes and in output control.

V.7 Installation

Three makefiles for different operating systems are included in the distribution as now indicated: DEC ULTRIX, IBM AIX, and SUN OS. If NESTLE is to be executed under either of these operating systems, just execute the appropriate makefile as indicated by the makefile tag *(i.e.* makefile.xxx where 'xxx' indicates the operating system). Execution on other operating systems will require modifications to the makefile.

MAIN

|

INPDATA GTIME POINTER GEOMETRY INPEDIT CONVER GASCATC GASCATH INITAL DEPLETE XSFDBK STARTER STEADYN OUTPUTSS SLOWTRAN ADJOINT OUTPUTAD TRANSIT

ADJOINT

|

XSFDADJ TRIDIA0 OUTINADJ OUTPCRT

AVGEDIT

|

RSTRING

BURNNODE

|

CALACV CHAIN

DEPLETE

|

MICROXNT BURNNODE XSECBU

INITAL

|

FLUIDCON

Figure 5: Dependence diagram of the NESTLE code

INIVAL

|

ECHOINP

INPDATA

|

POINTER FILE_CNT FILE_GEO FILE_XSC FILE_KIN FILE_PRF CHECK INIVAL

INPEDIT

|

INIVAL

LSORB

|

SORCE TRIDIA

LSORB0

|

SETUP0

MFST

|

SPECSHFT RELPOWER THFDBKK THFDBKS XSFDBK SCALAPRX SHAPECOR AMF SCALEXCT SCALING
LAMDASUB

Figure 5 (cont): Dependence diagram of the NESTLE code

MICROXNT

|

PINTER

NONNEMC

|

NONNETC NONPLMC NONONEC NONTWOC

NONNEMH

|

NONNETH NONPLMH NONONEH NONTWOH

NONONEC

|

DIRECT4 DIRECT8B DIR4FULL DIR2FULL

NONONEH

|

DIRECT2X2 DIR4FULL DIR2FULL DIRECT4 DIRECT8B

NONTWOC

|

DIRECT4 DIRECT8B DIRECT8 DIRECT16

Figure 5 (cont): Dependence diagram of the NESTLE code

NONTWOH

|

DIRECT2X2 DIRECT8 DIRECT16

OUTIN

|

GTIME LSORB CHEBY1 RELPOWER SXENON KSEARCH CNTROD PSEARCH NONNEMC NONNEMH THFDBKS
DEPLETE MFST RELPOWER SFST XSFDBK TRIDIA0

OUTINADJ

|

GTIME LSORB CHEBY1

OUTINTR

|

GTIME LSORB CHEBYTR NONNEMC NONNEMH RELPOWER THFDBKK MFST SFST XSFDBK TRIDIA0

OUTPUTAD

|

OUTPOINT

OUTPUTSS

|

AVGEDIT OUTPOINT

Figure 5 (cont): Dependence diagram of the NESTLE code

OUTPUTTR

|

AVGEDIT OUTPOINT

PERTURB

|

LINEAR

SETUP0

|

SORCE0 TRIDIA

SFST

|

AMF SCALEXCT SCALING RELPOWER

SLOWTRAN

|

PINTER XSFDBK OUTPCRT RELPOWER THFDBK TXENON CNTROD TRIDIA0 LSORB0 OUTIN PEAK OUTPCRT OUTPUTSS

STEADYN

|

PINTER XSFDBK NORM NORMFSP S XENON RELPOWER THFDBKS CNTROD TRIDIA0 LSORB0 OUTIN PEAK OUTPCRT

Figure 5 (cont): Dependence diagram of the NESTLE code

THFDBKK

|

FLUIDCON

THFDBKS

|

FLUIDCON

TRANSIT

|

PERTURB NORM NORMFSP RELPOWER STARTER PRECR CNTROD TXENON THFDBKK XSFDBK TRIDIA0
OUTINTR RELPOWER DECAYHN UPDATE PEAK OUTPCRT OUTPUTTR

XSECBU

|

PINTER

Figure 5 (cont): Dependence diagram of the NESTLE code

SUBROUTINE ADJOINT
THIS SUBROUTINE CONTROLS THE CALCULATION OF THE ADJOINT FLUX

SUBROUTINE AMF
THIS SUBROUTINE DETERMINES (A-F)*FLUX FOR THE FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE AVGEDIT
THIS SUBROUTINE EDITS OUT RADIAL AND AXIAL CORE AVERAGED PROPERTIES

SUBROUTINE BURNNODE
THIS SUBROUTINE SOLVES THE ISOTOPIC DEPLETION EQUATIONS

SUBROUTINE CALACV
THIS SUBROUTINE DETERMINES THE INTERACTION RATES REQUIRED TO SOLVE THE ISOTOPIC DEPLETION EQUATIONS

SUBROUTINE CHEBY1
THIS SUBROUTINE APPLIES THE SEMI-IMPLICIT CHEBYSHEV POLYNOMIAL ACCELERATION METHOD AND CHECKS FOR CONVERGENCE FOR THE STEADY-STATE PROBLEM

SUBROUTINE CHEBYTR
THIS SUBROUTINE APPLIES THE SEMI-IMPLICIT CHEBYSHEV POLYNOMIAL ACCELERATION METHOD AND CHECKS FOR CONVERGENCE FOR THE TRANSIENT PROBLEM

SUBROUTINE CHECK
THIS SUBROUTINE RECOGNIZES THE DIFFERENT NON-PERMISSIBLE RUNS AND FLAGS BACK AN ERROR MESSAGE ALONG WITH SUGGESTION FOR THE ALTERNATIVE RUN CASE

SUBROUTINE CNTROD
THIS SUBROUTINE DETERMINES THE FRACTION OF CONTROL GROUP INSERTED

SUBROUTINE CONVER
THIS SUBROUTINE CONVERTS THE INPUT DATA UNITS TO NESTLE'S INTERNAL WORKING UNITS

SUBROUTINE DECAYHN
THIS SUBROUTINE SOLVES THE DECAY HEAT PRECURSOR EQUATIONS

SUBROUTINE DEPLETE
THIS SUBROUTINE CONTROLS THE DEPLETION, DETERMINING NUMBER DENSITIES FOR THE MICROSCOPIC MODEL AND RETURNING THE MACROSCOPIC CROSS-SECTION EXPANSION COEFFICIENTS

SUBROUTINE DIR2FULL
THIS SUBROUTINE ANALYTICALLY SOLVES A 2X2 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ALL UNKNOWNS ARE EVALUATED. FULL MATRIX STRUCTURE ASSUMED IN ANALYTIC SOLUTION

Table 2: Listing of procedures and their functions

SUBROUTINE DIR4FULL
THIS SUBROUTINE ANALYTICALLY SOLVES A 4X4 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ALL UNKNOWNS ARE EVALUATED. FULL MATRIX ASSUMED IN ANALYTIC SOLUTION

SUBROUTINE DIRECT2X2
THIS SUBROUTINE ANALYTICALLY SOLVES A 2X2 MATRIX SYSTEM. NO VECTOR ARGUMENTS ARE EMPLOYED. ALL UNKNOWNS ARE EVALUATED. FULL MATRIX STRUCTURE ASSUMED IN ANALYTIC SOLUTION

SUBROUTINE DIRECT16
THIS SUBROUTINE ANALYTICALLY SOLVES A 16X16 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ONLY HALF OF UNKNOWNS ARE EVALUATED. MATRIX SPARSITY IS TAKEN ADVANTAGE OF IN ANALYTIC SOLUTION

SUBROUTINE DIRECT4
THIS SUBROUTINE ANALYTICALLY SOLVES A 4X4 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ALL UNKNOWNS ARE EVALUATED. MATRIX SPARSITY IS TAKEN ADVANTAGE OF IN ANALYTIC SOLUTION

SUBROUTINE DIRECT8
THIS SUBROUTINE ANALYTICALLY SOLVES A 8X8 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ONLY HALF OF UNKNOWNS ARE EVALUATED. MATRIX SPARSITY IS TAKEN ADVANTAGE OF IN ANALYTIC SOLUTION

SUBROUTINE DIRECT8B
THIS SUBROUTINE ANALYTICALLY SOLVES A 8X8 MATRIX SYSTEM. VECTOR ARGUMENTS ARE EMPLOYED. ALL UNKNOWNS ARE EVALUATED. MATRIX SPARSITY IS TAKEN ADVANTAGE OF IN ANALYTIC SOLUTION

SUBROUTINE ECHOINP
THIS SUBROUTINE ECHOES OUT THE RADIAL INPUT FIGURES

SUBROUTINE FILE_CNT
THIS SUBROUTINE READS THE GENERAL CONTROL INPUT PARAMETERS

SUBROUTINE FILE_GEO
THIS SUBROUTINE READS THE GEOMETRICAL PARAMETERS

SUBROUTINE FILE_KIN
THIS SUBROUTINE READS THE KINETIC PARAMETERS REQUIRED FOR TRANSIENT RUNS

SUBROUTINE FILE_PRF
THIS SUBROUTINE READS THE PARAMETERS USED TO CONTROL THE SOLUTION METHODS EMPLOYED AND PROVIDES THE CONVERGENCE CRITERIA

SUBROUTINE FILE_XSC
THIS SUBROUTINE READS THE CROSS-SECTION & T-H INPUT PARAMETERS

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE FLUIDCON
THIS SUBROUTINE CALCULATES COOLANT TEMPERATURE, DENSITY & VOID FRACTION BASED UPON INTERNAL ENERGY

SUBROUTINE GASCATC
THIS SUBROUTINE IS USED FOR THE CART GEOMETRY ONLY. IT CALCULATES THE NEEDED PARAMETERS TO GATHER THE NODES WITH THE SAME COLOR TOGETHER (I.E. BLACKS WITH BLACKS...ETC). FOR CART GEOMETRY, TWO DIFFERENT COLORS ARE USED. THIS WILL BE UTILIZED IN THE SOLUTION OF THE FINITE DIFFERENCE EQUATIONS

SUBROUTINE GASCATH
THIS SUBROUTINE IS USED FOR THE HEX GEOMETRY ONLY. IT CALCULATES THE NEEDED PARAMETERS TO GATHER THE NODES WITH SAME COLOR TOGETHER (I.E. BLACKS WITH BLACKS...ETC). FOR HEX GEOMETRY, THREE DIFFERENT COLORS ARE USED. THIS WILL BE UTILIZED IN THE SOLUTION OF THE FINITE DIFFERENCE EQUATIONS

SUBROUTINE GEOMETRY
THIS SUBROUTINE SETS UP THE GEOMETRY INCLUDING B.C., MESH EXPANSION, AND BUNDLE BOUNDARY IDENTIFICATION FOR OUTPUT CONTROL

FUNCTION GTIME
THIS SUBROUTINE RETURNS BACK THE ELAPSED TIME IN [SECONDS] APPLICABLE TO ULTRIX OR AIX OPERATING SYSTEMS MACHINES: DECStation 5000 OR IBM RS-6000 WORKSTATIONS

SUBROUTINE INITAL
THIS SUBROUTINE INITIALIZES THE FLUX, FISSION SOURCE & T-H CONDITIONS GUESSES, OR FOR RESTART OPTION ON READS IN THE RESTART FILE

SUBROUTINE INIVAL
THIS SUBROUTINE EDITS OUT THE INITIAL INPUT DATA, I.E.,CROSS SECTION DATA, GEOMETRY DATA AND CONTROL OPTIONS, IF USERS CHOOSE LONG EDITS FOR THE INPUT (AL3 = 'Y')

SUBROUTINE INPDATA
THIS SUBROUTINE CONTROLS THE OVERALL INPUT READING

SUBROUTINE INPEDIT
THIS SUBROUTINE EDITS OUT INPUT ASSOCIATED WITH THE TRANSIENT CASE, METHOD OF SOLUTION CONTROL, CONTROL OPTIONS, I/O FILE NAMES, AND IF ELECTED CROSS-SECTIONS AND T-H PARAMETERS

SUBROUTINE KSEARCH
THIS SUBROUTINE PERFORMS CRITICALITY SEARCH, ON FOUR DIFFERENT PARAMETERS: SOLUBLE POISON, INLET COOLANT TEMPERATURE, POWER LEVEL AND CONTROL BANK INSERTION

SUBROUTINE LAMDASUB
THIS SUBROUTINE DETERMINES THE SCALE FACTOR FOR THE FIXED-SOURCE SCALE FACTOR METHOD

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE LINEAR
THIS SUBROUTINE COMPLETES LINEAR INTERPOLATION OR EXTRAPOLATION

SUBROUTINE LSORB
THIS SUBROUTINE CALLS THE FOLLOWING TWO SUBROUTINES: SORCE: WHICH CALCULATES THE RHS OF THE FINITE DIFFERENCE EQUATIONS FOR A SPECIFIC COLOR AND TRIDIA: WHICH SOLVES FOR THE FLUX BY SOLVING A TRIDIAGONAL SYSTEM OF EQUATIONS FOR A PARTICULAR COLOR. THE FLUX IS ALSO ACCELERATED USING THE OMEGAS WHICH WERE PRE-CALCULATED IN SUBROUTINE LSORB0

SUBROUTINE LSORB0
THIS SUBROUTINE CALCULATES THE NUMBER OF INNERS PER OUTER AND THE ACCELERATION PARAMETERS FOR THE COLOR LINE SOR EQUATIONS

PROGRAM MAIN
THIS PROGRAM IS THE MAIN PROGRAM FOR NESTLE

SUBROUTINE MFST
THIS SUBROUTINE CONTROLS THE MULTI-FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE MICROXNT
THIS SUBROUTINE DETERMINES THE MICROSCOPIC CROSS-SECTION EXPANSION COEFFICIENTS FOR THE DEPLETABLE ISOTOPES

FUNCTION NEWPAGE
FUNCTION STARTS A NEW PAGE OF OUTPUT

SUBROUTINE NONNEMC
THIS SUBROUTINE SOLVES THE NEM EQUATIONS.
********** FOR CARTESIAN GEOMETRY **********
THE NODAL METHOD USED HERE IS NON-LINEAR NEM. WE SOLVE A ONE OR TWO NODE PROBLEM FOR EVERY INTERFACE IN THE CORE TO UPDATE THE COUPLING COEFFICIENTS

SUBROUTINE NONNEMH
THIS SUBROUTINE SOLVES THE NEM EQUATIONS.
********** FOR HEXAGONAL GEOMETRY **********
THE NODAL METHOD USED HERE IS NON-LINEAR NEM. WE SOLVED A ONE OR TWO NODE PROBLEM FOR EVERY INTERFACE IN THE CORE TO UPDATE THE COUPLING COEFFICIENTS

SUBROUTINE NONNETC
THIS SUBROUTINE DETERMINES THE CURRENTS FOR NEM
********** FOR CARTESIAN GEOMETRY **********

SUBROUTINE NONNETH
THIS SUBROUTINE DETERMINES THE CURRENTS FOR NEM
********** FOR HEXAGONAL GEOMETRY **********

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE NONONEC
THIS ROUTINE SOLVES A ONE-NODE PROBLEM. IT IS USED ONLY FOR EDGE NODES WITHOUT ZERO CURRENT BOUNDARY CONDITION WHEN UPDATING THE NET CURRENTS BY NONNEM.
********** FOR CARTESIAN GEOMETRY **********

SUBROUTINE NONONEH
THIS ROUTINE SOLVES A ONE-NODE PROBLEM. IT IS USED ONLY FOR EDGE NODES WITHOUT A ZERO CURRENT BOUNDARY CONDITION WHEN UPDATING THE NET CURRENTS BY NONNEM.
********** FOR HEXAGONAL GEOMETRY **********

SUBROUTINE NONPLMC
THIS SUBROUTINE CALCULATES THE LEAKAGES FOR EVERY NODE, IN EVERY DIRECTION, AND THE EXPANSION COEFFICIENTS FOR THE QUADRATIC LEAKAGE APPROXIMATION.
********** FOR CARTESIAN GEOMETRY **********

SUBROUTINE NONPLMH
THIS SUBROUTINE CALCULATES THE LEAKAGES FOR EVERY NODE, IN EVERY DIRECTION, AND THE EXPANSION COEFFICIENTS FOR THE QUADRATIC LEAKAGE APPROXIMATION.
********** FOR HEXAGONAL GEOMETRY **********

SUBROUTINE NONTWOC
THIS SUBROUTINE SOLVES THE TWO-NODE PROBLEM AND RETURNS THE NEM CURRENT 'JNEM'.
********** FOR CARTESIAN GEOMETRY **********
THIS SUBROUTINE FILLS IN THE 16X16 (FOR 2 GROUPS) OR 32X32 (FOR 4 GROUPS) MATRIX OF THE 2-NODE PROBLEM, AND CALLS THE ANALYTIC SOLVERS WHICH ARE DIFFERENT REGARDING THE NUMBER OF GROUPS. WE HAVE TAKEN ADVANTAGE OF THE REDUCIBILITY OF THE MATRIX. FOR ONE 2-NODE PROBLEM, INVOLVING SAY NODE L AND L+1, THE EVEN EXPANSION COEFFICIENTS FOR THE FLUX OF NODE L+1 ARE THE SAME AS FOR THE ONES FOR THE 2-NODE PROBLEM INVOLVING NODE L+1 AND L+2. SO WE DON'T HAVE TO SOLVE FOR THEM TWICE

SUBROUTINE NONTWOH
THIS SUBROUTINE SOLVES THE TWO-NODE PROBLEM AND RETURNS THE NEM CURRENT'JNEM'.
********** FOR HEXAGONAL GEOMETRY **********
THIS SUBROUTINE FILLS IN THE 16X16 (FOR 2 GROUPS) OR 32X32 (FOR 4 GROUPS) MATRIX OF THE 2-NODE PROBLEM, AND CALLS THE ANALYTIC SOLVERS WHICH ARE DIFFERENT REGARDING THE NUMBER OF GROUPS. WE HAVE TAKEN ADVANTAGE OF THE REDUCIBILITY OF THE MATRIX. FOR ONE 2-NODE PROBLEM, INVOLVING SAY NODE L AND L+1, THE EVEN EXPANSION COEFFICIENTS FOR THE FLUX OF NODE L+1 ARE THE SAME AS FOR THE ONES FOR THE 2-NODE PROBLEM INVOLVING NODE L+1 AND L+2. SO WE DON'T HAVE TO SOLVE FOR THEM TWICE!!

SUBROUTINE NORM
THIS SUBROUTINE NORMALIZES THE FLUX AND FISSION SOURCE TO A CORE RELATIVE AVERAGE POWER = 1.

SUBROUTINE NORMFSP
THIS SUBROUTINE NORMALIZES THE FLUX AND FISSION SOURCE TO AN INPUT SPECIFIED SCALING

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE OUTIN
THIS SUBROUTINE PERFORMS OUTER-INNER ITERATIONS UTILIZING FDM OR NEM OPTION FOR STEADY-STATE SOLUTION

SUBROUTINE OUTINADJ
THIS SUBROUTINE PERFORMS OUTER-INNER ITERATIONS UTILIZING FDM OR NEM OPTION FOR THE ADJOINT SOLUTION

SUBROUTINE OUTINTR
THIS SUBROUTINE PERFORMS OUTER-INNER ITERATIONS UTILIZING FDM OR NEM OPTION FOR TRANSIENT SOLUTION

SUBROUTINE OUTPCRT
THIS SUBROUTINE OUTPUTS VALUES TO THE SCREEN (I.E. CRT)

SUBROUTINE OUTPOINT
THIS SUBROUTINE OUTPUTS POINT-WISE VALUES OF PARAMETER PASSED THROUGH CALLING ARGUMENT LIST

SUBROUTINE OUTPUTAD
THIS SUBROUTINE OUTPUTS THE ADJOINT PROBLEM'S SOLUTION

SUBROUTINE OUTPUTSS
THIS SUBROUTINE OUTPUTS THE STEADY-STATE PROBLEM'S SOLUTION

SUBROUTINE OUTPUTTR
THIS SUBROUTINE OUTPUTS THE TRANSIENT PROBLEM'S SOLUTION

SUBROUTINE PEAK
THIS SUBROUTINE DETERMINES THE TOTAL PEAKING FACTOR AND LOCATION

SUBROUTINE PERTURB
THIS SUBROUTINE INTERPOLATES THE TIME DEPENDENT INPUT PARAMETERS FOR THE TRANSIENT PROBLEM

SUBROUTINE PINTER
THIS SUBROUTINE COMPLETES QUADRATIC INTERPOLATION (OR EXTRAPOLATION) USING LAGRANGIAN INTERPOLATION POLYNOMIAL

SUBROUTINE POINTER
THIS SUBROUTINE DETERMINES THE A ARRAY POINTERS (I.E. STARTING LOCATIONS OF ARRAYS)

SUBROUTINE PRECR
THIS SUBROUTINE SOLVES THE DELAYED NEUTRONS PRECURSOR EQUATIONS

FUNCTION PROPPOLY
THIS SUBROUTINE CALCULATES SPECIFIED PROPERTY AS FUNCTION OF A STATED DEPENDENCE (PARAM)

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE PSEARCH
THIS SUBROUTINE PERFORMS POWER LEVEL SEARCH, ON THREE DIFFERENT PARAMETERS: SOLUBLE POISON, INLET COOLANT TEMPERATURE AND CONTROL BANK INSERTION

SUBROUTINE RELPOWER
THIS SUBROUTINE DETERMINES THE TOTAL CORE POWER LEVEL ACCOUNTING FOR DECAY HEAT AND SCALES THE POWER DENSITY TO A RELATIVE CORE POWER LEVEL = 1.

SUBROUTINE RSTRING
THIS SUBROUTINE LOADS A NUMERIC VALUE INTO AN ALPHANUMERIC VARIABLE

SUBROUTINE SCALAPRX
THIS SUBROUTINE CALCULATES THE RATIO OF THE EFFECTIVE EXTERNAL SOURCE TO THE FISSION SOURCE USED BY THE FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE SCALEXCT
THIS SUBROUTINE DETERMINES THE RATIO OF THE EFFECTIVE EXTERNAL SOURCE TO (A-F)*FLUX TO GET SCALE FACTOR UPDATE

SUBROUTINE SCALING
THIS SUBROUTINE SCALES THE FLUX AND FISSION SOURCE USING THE SCALE FACTOR FROM THE FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE SETUP0
THIS SUBROUTINE SOLVES THE HOMOGENOUS PROBLEM USING COLOR LINE G-S IN SUPPORT OF DETERMINING OPTIMUM RELAXATION PARAMETERS AND NUMBER OF INNER ITERATIONS PER OUTER ITERATION

SUBROUTINE SFST
THIS SUBROUTINE PERFORMS THE SINGLE FIXED-SOURCE SCALING TECHNIQUE PROCEDURE FOR THE FSP STEADY-STATE CASE

SUBROUTINE SHAPECOR
THIS SUBROUTINE ADJUSTS THE FLUX FOR COOLANT SPECTRAL EFFECTS WITHIN THE FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE SLOWTRAN
THIS SUBROUTINE PROVIDES OVERALL CONTROL OF THE TRANSIENT FISSION PRODUCT PROBLEM

SUBROUTINE SORCE
THIS SUBROUTINE CALCULATES THE RHS FOR THE TRIDIAGONAL SYSTEM TO BE SOLVED BY SUBROUTINE TRIDIA. THE TRIDIAGONAL SYSTEM RESULTS FROM USING THE COLOR LINE SOR METHOD WHICH IS USED TO SOLVE THE FINITE DIFFERENCE FORM OF THE DIFFUSION EQUATIONS. THE RHS HERE INCLUDES FISSION, SCATTERING, AND DIFFUSION TERMS

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE SORCE0
THIS SUBROUTINE CALCULATES THE RHS FOR THE TRIDIAGONAL SYSTEM TO BE SOLVED BY SUBROUTINE TRIDIA. THE EQUATION TO BE SOLVED IS A*PHI=0, WHERE A IS THE COEFFICIENT MATRIX AND PHI IS THE FLUX. THE RHS HERE DOES NOT INCLUDE ANY FISSION OR SCATTERING. IT ONLY INCLUDES DIFFUSION TERMS. IT IS USED IN OBTAINING THE COLOR LINE SOR EXTRAPOLATION PARAMETERS AND NUMBER OF INNER ITERATIONS PER OUTER ITERATION

SUBROUTINE SPECSHFT
THIS SUBROUTINE DETERMINES THE B**2 VALUES USED IN MAKING THE COOLANT SPECTRAL SHIFT CORRECTION IN THE FIXED-SOURCE SCALE FACTOR METHOD

SUBROUTINE STARTER
THIS SUBROUTINE SETS INITIAL VALUES TO INITIATE THE TRANSIENT FROM

SUBROUTINE STEADYN
THIS SUBROUTINE PROVIDES OVERALL CONTROL OF THE STEADY-STATE SOLUTION

SUBROUTINE SXENON
THIS SUBROUTINE SOLVES FOR THE STEADY-STATE NUMBER DENSITIES OF THE TRANSIENT FISSION PRODUCTS

SUBROUTINE THFDBKK
THIS SUBROUTINE CALCULATES COOLANT INTERNAL ENERGY, COOLANT DENSITY AND FUEL TEMPERATURES FOR THE TRANSIENT PROBLEM

SUBROUTINE THFDBKS
THIS SUBROUTINE CALCULATES COOLANT INTERNAL ENERGY, COOLANT DENSITY AND FUEL TEMPERATURES FOR THE STEADY-STATE PROBLEM

SUBROUTINE TRANSIT
THIS SUBROUTINE PROVIDES OVERALL CONTROL OF THE TRANSIENT PROBLEM

SUBROUTINE TRIDIA
THIS SUBROUTINE SOLVES A TRIDIAGONAL SYSTEM OF EQUATIONS WHICH RESULTS FROM USING THE COLOR LINE SOR METHOD. IT SOLVES FOR THE FLUX OF A PARTICULAR COLOR EACH TIME IT IS CALLED

SUBROUTINE TRIDIA0
THIS SUBROUTINE FACTORS THE TRIDIAGONAL MATRICES ASSOCIATED WITH COLOR LINE SOR AND ASSIGNS TO ARRAYS FOR EACH COLOR

SUBROUTINE TXENON
THIS SUBROUTINE SOLVES FOR THE TRANSIENT NUMBER DENSITIES OF THE TRANSIENT FISSION PRODUCTS

SUBROUTINE UPDATE
THIS SUBROUTINE SAVES TIME-STEP VALUES OF VARIOUS PARAMETERS FOR USAGE IN THE NEXT TIME-STEP SOLUTION

Table 2 (cont): Listing of procedures and their functions

SUBROUTINE XSECBU
THIS SUBROUTINE DETERMINES THE NUCLEAR PROPERTIES (E.G. CROSS-SECTION EXPANSION COEFFICIENTS) AT THE NODE COLOR AND BURNUP, EXCEPT FOR THE DEPLETABLE ISOTOPES MODELED USING THE MICROSCOPIC OPTION

FUNCTION XSECPOLY
THIS SUBROUTINE CALCULATES X-SECTIONS ACCOUNTING FOR COOLANT TEMPERATURE, COOLANT DENSITY, FUEL TEMPERATURE AND SOLUBLE POISON FEEDBACK CORRECTIONS

FUNCTION XSECPOLY2
THIS SUBROUTINE CALCULATES X-SECTIONS ACCOUNTING FOR COOLANT TEMPERATURE, COOLANT DENSITY, FUEL TEMPERATURE AND SOLUBLE POISON FEEDBACK CORRECTIONS. DIFFERENCE WITH XSECPOLY IS DIMENSION OF CALLING ARGUMENT ARRAYS

SUBROUTINE XSFDADJ
THIS SUBROUTINE DETERMINES THE MATRIX TRANSPOSE OF THE COEFFICIENT MATRIX REQUIRED FOR THE ADJOINT FLUX SOLUTION

SUBROUTINE XSFDBK
THIS SUBROUTINE DETERMINES THE MACROSCOPIC AND MICROSCOPIC CROSS-SECTIONS AND OTHER NEUTRONIC NODE VALUES AND USES THEM IN DETERMINING THE COEFFICIENT MATRIX

Table 2 (cont): Listing of procedures and their functions

```
NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE
NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE
NESTLE                                                          NESTLE
NESTLE                                                          NESTLE
NESTLE                        VERSION 1.0                       NESTLE
NESTLE                  ELECTRIC POWER RESEARCH CENTER          NESTLE
NESTLE                  NORTH CAROLINA STATE UNIVERSITY         NESTLE
NESTLE                  COPYRIGHT 1994 - BY NCSU                NESTLE
NESTLE                                                          NESTLE
NESTLE                  NESTLE DICTIONARY PROGRAM               NESTLE
NESTLE                                                          NESTLE
NESTLE                                                          NESTLE
NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE
NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE NESTLE
```

WHICH OPTION DO YOU WISH TO ENABLE
(E=EXIT, F=FIND, D=DELETE, A=ADD)? **F**

WHAT IS THE VARIABLE NAME THAT YOU WISH THE
DEFINITION FOUND FOR? **SW**

DEFINITION:
MESH DEPENDENT FLUX [SOLUTION TO THE PROBLEM] / EIGENVECTOR FOR THE EIGNVALUE PROBLEM

WHICH OPTION DO YOU WISH TO ENABLE
(E=EXIT, F=FIND, D=DELETE, A=ADD)? **A**

WHAT IS THE VARIALBE NAME YOU WISH TO ADD? TEST1

INPUT DEFINITION FOR THE VARIABLE:
JUST A TEST VARIABLE FOR THE DICTIONARY PROGRAM

WHICH OPTION DO YOU WISH TO ENABLE
(E=EXIT, F=FIND, D=DELETE, A=ADD)? **F**

WHAT IS THE VARIABLE NAME THAT YOU WISH THE
DEFINITION FOUND FOR? **TEST1**

DEFINITION:
JUST A TEST VARIABLE FOR THE DICTIONARY PROGRAM

Table 3: Sample interactive session with NESTLE.DICT

WHICH OPTION DO YOU WISH TO ENABLE
(E=EXIT, F=FIND, D=DELETE, A=ADD)? **D**

WHAT IS THE VARIABLE NAME THAT YOU WISH THE
DEFINITION DELETED FOR? **TEST1**

ARE YOU CERTAIN THAT YOU WANT THE VARIABLE
DEFINITION DELETED (Y,N)? **Y**

VARIABLE DEFINITION DELETED.

WHICH OPTION DO YOU WISH TO ENABLE
(E=EXIT, F=FIND, D=DELETE, A=ADD)? **E**

IF VARIABLE NAMES AND DEFINITIONS HAVE BEEN ADDED
OR DELETED, DO YOU WANT TO SAVE CHANGES IN DATA BASE (Y,N)? **Y**

Table 3 (cont): Sample interactive session with NESTLE.DICT

adf.fcb	crhs.fcb	nline.fcb	spectral.fcb	tim.fcb
adj.fcb	crit.fcb	nonfue.fcb	start.fcb	time.fcb
array.fcb	crod.fcb	nterm.fcb	temp1.fcb	time1.fcb
basic.fcb	dataf.fcb	numsurf.fcb	temp11.fcb	timetr.fcb
bcs.fcb	depletepc.fcb	only.fcb	temp12.fcb	varlen.fcb
bcshex.fcb	extsor.fcb	opti.fcb	temp13.fcb	veloc.fcb
buckl.fcb	flamdold.fcb	outlong.fcb	temp6.fcb	xeopt.fcb
burn.fcb	fpxs.fcb	param.fcb	temp7.fcb	xsec1.fcb
bypass.fcb	gasch.fcb	perttr.fcb	temp9.fcb	xsec2.fcb
che.fcb	geom.fcb	pertv.fcb	th_cof.fcb	xspolycom.fcb
cheby.fcb	hexdim.fcb	power.fcb	thcoef.fcb	
cntl.fcb	hgeo.fcb	restinp.fcb	thermk.fcb	
conv.fcb	multit.fcb	restotp.fcb	thermo.fcb	
convfact.fcb	nemcnt.fcb	soln2.fcb	thmargin.fcb	

Table 4: Listing of fcb files containing named COMMON blocks