SKETCH-N: A NODAL NEUTRON DIFFUSION CODE FOR SOLVING STEADY-STATE AND KINETICS PROBLEMS

VOL. I. MODEL DESCRIPTION

Vyacheslav G. Zimin,
Vladislav. I. Romanenko

July 30, 2025

Contents

List of Figures

List of Tables

Abstract

Program Name and Title

SKETCH-N: A nodal code for solving neutron diffusion equations of steady-state and kinetics problems.

Simulation scope

The SKETCH-N code solves neutron diffusion equations in x-y-z and hexagonal-z geometries for steady-state and neutron kinetics problems. The code can treat an arbitrary number of neutron energy groups and delayed neutron precursors.

Methods used

The polynomial, semi-analytic and analytic nodal methods based on the nonlinear iteration procedure can be used for spatial discretization of diffusion equations. The time integration of the neutron kinetics problem is performed by a fully implicit scheme with an analytical treatment of the delayed neutron precursors. The steady-state eigenvalue problems are solved by inverse iterations with Wielandt shift, the Chebyshev adaptive acceleration procedure is used for the neutron kinetics problems. The block symmetric Gauss-Seidel preconditioner is applied in the both iterative methods. The flux-weighting homogenization procedure is used for partially-rodded nodes to minimize a rod cusping effect. Simple one-phase model of the thermal-hydraulics of fuel assembly is included in the code. The code also has an interface module for a coupling with transient analysis codes , such as TRAC. The interface module performs a data exchange between the codes, synchronizes a time stepping and maps the neutronics data onto thermal-hydraulics spatial mesh and vice versa. The interface module is based on the message passing library PVM (Parallel Virtual Machine).

Limitations

The code can treat the neutron diffusion problems in Cartesian geometry. Few-

group macro cross sections and their dependencies are provided by a code user. The code does not have fuel burn-up modelling capabilities. An external thermal-hydraulics code is generally required for the calculation of the "real-life" problems.

Typical Running Time

The running time of the full-core case C1 of the PWR NEACRP rod ejection benchmark (2 neutron energy groups, 6 groups of the delayed neutron precursors, 884x18 neutronics nodes, 910 time steps) is 68 minutes on Sun UltraSPARC I (143 MHz) with an internal thermal hydraulics model.

Code features

Dimensions of a problem are specified as parameters in the include files, the code should be recompiled when the problem dimensions are changed. The code has PVM- based interface module developed for a coupling with transient thermal-hydraulics codes. The interface model has been used for a coupling of the SKETCH-N code with the J-TRAC (TRAC-PF1) and TRAC-BF1 codes.

Related and Auxiliary Programs

PVM library is used for the interface module of the code.

Status

The SKETCH-N code has been verified by solving the steady-state and neutron kinetics benchmark problems. The coupled J-TRAC/SKETCH-N code system has been verified against NEACRP PWR rod ejection and rod withdrawal benchmarks. NEACRP BWR cold water injection benchmark has been used for verification of the TRAC-BF1/SKETCH-N system.

References

Zimin V. G. "Nodal Neutron Kinetics Models Based on Nonlinear Iteration Procedure for LWR Analysis", PhD Thesis, Research Laboratory for Nuclear Reactors, Tokyo Institute of Technology, August 1997.

Zimin, V.G., and H. Ninokata, "Nodal Neutron Kinetics Model Based on Nonlinear Iteration Procedure for LWR Analysis," Ann. Nucl. Energy, 25, 507-528, 1998

Zimin, V.G., H. Ninokata, and L. R. Pogosbekyan "Polynomial and Semi-Analytic Nodal Methods for Nonlinear Iteration Procedure," Proc. of the Int. Conf. on the Physics of Nuclear Science and Technology, October 5-8, 1998, Long Island, New York, American Nuclear Society, vol. 2, pp. 994- 1002, 1998

Zimin V. G., H. Asaka, Y. Anoda, and M. Enomoto, "Verification of the J-TRAC code with 3D Neutron Kinetics Model SKETCH-N for PWR Rod Ejection Analysis", Proc. of the 9 International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH 9), San Francisco, California, October 3- 8, CD-ROM, 1999.

Asaka, H., V. G. Zimin, T. Iguchi, and Y. Anoda, "Coupling of the Thermal-Hydraulic TRAC Codes with 3D Neutron Kinetics Code SKETCH-N", Proc. of OECD/CSNI Workshop on Advanced Thermal- Hydraulic and Neutronics Codes: Current and Future Applications, Barcelona, Spain, 10-13 April, 2000

Zimin, V. G., H. Asaka, Y. Anoda, E. Kaloinen and R. Kyrki-Rajamaki, "Analysis of NEACRP 3D BWR Core Transient Benchmark", Proc. of the 4 Intl. Conf. on Supercomputing in Nuclear Application SNA 2000, September 4-7, 2000, Tokyo, Japan.

Machine Requirements

A workstation under UNIX.

Program Language Used

Fortran 77 and Fortran 90

Other Programming or Operating Information or Restrictions

The interface module requires PVM installed on a computer. The PVM is a public domain software available from NETLIB [http://www.epm.ornl.gov/pvm/pvm_home.html].

Material Available

Source Code

Sample LWR Benchmark Problems Input and Output Files

SKETCH-N Manual, vol. I. Model Description

SKETCH-N Manual, vol. II User's Guide

Keywords

kinetics, three-dimensional, neutron diffusion, nodal methods, nonlinear iteration procedure, reactor transient analysis.

Contacts

Vyacheslav G. Zimin

Division 836 "Laboratory of Simulator Systems", National Research Nuclear University "MEPhI", Kashirskoe shosse, 31, Moscow, 115409, Russia

Tel.: +7-963-648-87-81
E-mail: vgzimin@mail.ru

Acknowledgments

The finite-difference code SKETCH has been developed during my doctor study and work on the advanced fast breeder reactor analysis project at the Department of Theoretical and Experimental Reactor Physics of Moscow Engineering Physics Institute (MEPhI), Moscow, Russia in 1989-1994. I'm especially grateful to Dr. Nikolay V. Schukin of MEPhI for the encouragement and interest that he has contributed over the years. The nodal code SKETCH-N for LWR analysis has been developed during my doctor course study in the Research Laboratory of Nuclear Reactors of the Tokyo Institute of Technology, Tokyo, Japan in 1994-1997. Thanks a lot to the supervisor of my doctor thesis - Prof. Hisashi Ninokata for all time, which he spent teaching me, his tolerance, kindness and hospitality. During my doctor study I had very helpful and stimulating discussions with Mr. Hideaki Ikeda and Mr. Akitoshi Hotta, both of Toden Software and Prof. Toshikazu Takeda of Osaka University. An interface module for the coupling with the transient reactor analysis codes J-TRAC and TRAC-BF1 has been developed during my stay as a postdoctoral researcher at the Thermal Hydraulics Safety Research Laboratory, Department of Reactor Safety Research, Japan Atomic Energy Research Institute (JAERI) in 1997-2000. During this time numerous code improvements have been also done. The author wants to thank Mr. Hideaki Asaka and Dr. Yoshinari Anoda, both of JAERI for the interest to my work, help in many problems and excellent working conditions.

I also wish to thank Dr. Leonid R. Pogosbekyan, VNIIAES, Moscow, Russia for his contribution to the implementation of the analytical nodal method based on matrix function theory and other helpful and stimulating discussions.

The last but not the least thanks go to my wife Olya and to all my friends for their support and encouragement.

Chapter 1

Introduction

SKETCH-N is a FORTRAN computer code that solves few-group neutron diffusion equations in three- dimensional Cartesian and Hexagonal-z geometries. The code can treat the steady-state eigenvalue problem and neutron kinetics problems. A criticality search of the soluble boron concentration can be performed in the case of the eigenvalue problem.

Arbitrary number of neutron energy groups and delayed neutron precursors can be used. The code can treat upscattering. Reactor model should be described in three-dimensional Cartesian or Hexagonal-z geometry, arbitrary mesh size is allowed in all directions. Two and one-dimensional models can be also utilised. Reflective boundary conditions can be use to model a quarter- and half-symmetric cores. The boundary conditions include zero flux and current conditions and albedo type conditions.

Spatial discretization of the diffusion equations can be done using several nodal methods. All of them are based on the transverse integration procedure and quadratic leakage approximation. Polynomial nodal method (PNM) is based on the fourth order expansion of the transverse-integrated neutron flux into Legendre polynomials. In the semi-analytical nodal method (SANM), the third and fourth Legendre polynomials are replaced by hyperbolic sine and cosine. The sine and cosine are modified in the way to preserve an orthogonal system of the basic functions, similar to Legendre polynomials. An exact analytic solution of the transverse-integrated nodal equations is applied in the analytic nodal method

(ANM). The nodal equations are solved using the nonlinear iteration procedure. In this approach, the coarse-mesh finite-difference (CMFD) method is forced to match the results of the nodal method. The nodal equations are used only to compute the nodal coupling coefficients, which are introduced into the CMFD equations.

The resulting solution procedure is decoupled into two processes:

- iterative solution of the CMFD equations over the reactor domain with respect to the node-averaged neutron fluxes and eigenvalue;
- local solutions of the two-node problems for each face of the nodes to compute the face-averaged neutron currents and update the nodal coupling coefficients.

If the nodal coupling coefficients are set to zero we obtain the mesh-centered finite-difference method, which also can be used in the code. An application of the nonlinear iteration procedure results in an efficient algorithm because we need to solve relatively time-consuming nodal equations only a few times during the steady-state calculation and even a once per time step for the transient problems. An orthogonal system of the basic functions of the both PNM and SANM results in a very small and simple system of the nodal equations. In the both methods, for each two-node problem we need to solve only one G $\times$ G and one 2G $\times$ 2G system of equations, where G is a number of the neutron energy groups. ANM is implemented using even more efficient approach using the matrix function theory. In the case of two group calculations, when the eigenvalues of 2x2 buckling matrix are easy to compute, the nodal equation of ANM are reduced to two equations for the face-averaged neutron fluxes. As a result computing time of ANM can be practically the same as that of PNM in the case of two-group calculations. A code user can select the nodal method depending on the problem at hand. PNM provides maximum efficiency, while accuracy is usually better with SANM and ANM. In the case of the multigroup calculations, efficiency of ANM drastically decreases in a comparison with that of PNM and SANM. As default nodal method we recommend to use SANM because it is usually the most accurate method and it is the most stable to a choice of a mesh size. The steady-state neutron diffusion

equations, nonlinear iteration procedure and the nodal methods are described in Chapters 2 and 3.

Time discretization of the neutron kinetics equations is based on the fully-implicit scheme with an analytic integration of the delayed neutron precursors. The nodal equations of the neutron kinetics problem are reduced to the nodal equations of the steady-state eigenvalue problem. An automatic time step size control based on the time-step doubling technique is used in the code. As an option the SKETCH-N code has point kinetics model. Parameters of the point kinetics equations are computed from the macro cross sections at each time step. The point kinetics model is very useful to show the effect of the change of the neutron flux shape on the transient results. Neutron kinetics equations and the time- discretization techniques are given in Chapter 4.

A solution of the CMFD equations is performed by iterative methods. Three iterative levels are used in the case of the steady-state eigenvalue problem. At the top iterative level, called nonlinear iterations, macro cross sections, nodal coupling coefficients and matrices are recomputed depending on the values of thermal-hydraulics parameters, node-average neutron fluxes and eigenvalue. At the next level, called outer iterations, the linear generalized eigenvalue problem is solved by inverse iterations with Wieland shift accelerated by Chebyshev polynomials. A solution of the linear system of equations at each outer iteration is performed by inner iterations. The block symmetric Gauss-Seidel method is used at the inner iteration level, with a size of the block equal to a number of neutron energy groups. The iterative solution techniques applied in the steady-state problems are described in Chapter 5.

In the case of the neutron kinetics problem, the CMFD equations at each time step are linearized taking the values of thermal-hydraulics parameters and nodal coupling coefficients from the previous time step. The resulting linear system of equations is solved by the Chebyshev semi-iterative method. The Chebyshev iterative parameters are computed using the adaptive procedure based on a comparison of the theoretical and actual convergence rate. The block symmetric Gauss-Seidel method is used as a preconditioner. The iterative methods used in the neutron kinetics problems are discussed in Chapter 6.

Based on the value of the thermal-hydraulics parameters, the SKETCH-N code

updates the values of the macro cross sections for each reactor node. Two macro cross section models are available in the code. The first one is developed for the calculations of the NEACRP LWR core transient benchmark (Finnemann and Galati, 1992) and based on the polynomial representation of the macro cross section as functions of boron concentration, Doppler fuel temperature, temperature and density of coolant. The reactor is defined at the beginner of cycle 1, burnup dependencies and Xenon poisoning are not modelled. The second more general form of the macro cross sections is developed based on the data provided in NEA/NSC Ringhals- 1 Boiling Water Reactor (BWR) stability benchmark (Lefvert, 1994). In this problem, the cross section file contains:

- macro cross sections given as three-dimensional tables with entries for burnup, void and conversion history (conversion history stands as a collective label for void history and control rod history);
- data for the Doppler feedback, which are given as a change of the infinite multiplication factor depending on square root of the Doppler fuel temperature, void fraction and burnup;
- microscopic xenon thermal absorption cross section is specified as a polynomial function of the void and burnup.

A basic set of the cross sections is computed in the code by a tensor-product linear interpolation in the given three-dimensional table. The Doppler effect is incorporated into the fast-group absorption cross section estimated from the condition to preserve the change of the infinite multiplication factor. Xenon poisoning is simulated using the given form of the micro absorption cross section. The both macro cross section models are developed for the given macro cross section data and they are not general enough to cover all Light Water Reactor (LWR) transients. An application of the code to other transients or other reactor types will require a change of the cross section models. The other problem related to the calculation of the macro cross sections is homogenisation of the cross sections for the nodes with a partially inserted control rod. The problem is known as a rod cusping effect. The choice of the homogenisation method has strong impact on accuracy of the rod ejection and rod withdrawal transients in Pressurized

Water Reactor (PWR). In the SKETCH-N code, flux-weighting homogenisation procedure is used for partially rodded nodes. The calculation of the macro cross sections is discussed in Chapter 7. An internal thermal-hydraulics model can treat single-phase coolant flow under a constant pressure condition and developed mostly for the code benchmarking. Heat conduction equations are discretized using the finite-difference method. Only radial heat conduction is considered. Arbitrary mesh size is allowed. The fuel material properties are defined based on the NEACRP PWR rod ejection benchmark specification (Finnemann and Galati, 1992). The mass continuity and energy conservation equations for coolant are solved using the finite-difference method. A code user should define the mass flow rate for each channel used in the calculations. The thermal-hydraulics module of the code is presented in Chapter 8.

For analysis of the reactor transients with two-phase coolant flow SKETCH-N code should be coupled with an external thermal-hydraulics model. To simplify the code coupling procedure the SKETCH-N code has an interface module based on the message-passing library PVM. The interface module has been used for the coupling of the SKETCH-N code with the transient analysis codes J-TRAC (TRAC-PF1) for PWR applications and TRAC-BF1 for BWR analysis. The codes are treated as separate processes and they communicate sending and receiving messages. The interface module is responsible for the data exchange between the codes, mapping of the data on the code spatial meshes and synchronization of the time stepping. SKETCH-N sends the messages with computed power distribution and time step size and receives the messages with feedbacks and proposal for a new time step size. Feedbacks are Doppler fuel temperature, coolant temperature and coolant density or void. Data mapping technique is based on the mapping matrix approach providing flexibility for a code user in the selection of a number of fuel bundles per a thermal-hydraulics channel and axial mesh size. Time stepping of the coupled code system is based on the leap-frog semi-implicit scheme, SKETCH-N code performs the time step calculation first. Time step size of the code system is equal to a minimum of the time step sizes proposed by the codes. Our experience have shown that a coupling using the interface module can be done easy even with large and complicated codes, such as TRAC. The interface module is described in the Chapter 9.

Chapter 2

Nodal equations for steady-state problems in Cartesian geometry

2.1 Neutron diffusion equation

In Cartesian geometry few-group steady-state neutron diffusion equations in P form are written as

$$\sum_{u=x,y,z} \frac{\partial}{\partial u} j_{gx}(x,y,z) + \Sigma_{rg}(x,y,z)\varphi_g(x,y,z) = \sum_{\substack{g'=1 \\ g' \neq g}}^{G} \Sigma_{sgg'}(x,y,z)\varphi_{g'}(x,y,z) + \frac{1}{k_{eff}} \chi_g \sum_{g'=1}^{G} \nu\Sigma_{fg'}(x,y,z)\varphi_{g'}(x,y,z) \quad (2.1)$$

$$j_{gu}(x,y,z) = -D_g(x,y,z)\frac{\partial}{\partial u}\varphi_g(x,y,z), u = x,y,z; g = 1,...,G, \quad (2.2)$$

where

g - neutron energy group index;

G - total number of neutron energy groups;

$j_{gu}(x,y,z)$ - u-component of the net neutron current $[cm^{-2}s^{-1}]$;

$\varphi_g(r)$ - neutron flux $[cm^{-2}s^{-1}]$;

$\Sigma_{rg}(r)$ - macroscopic removal cross section $[cm^{-1}]$;

$\Sigma_{sgg'}(r)$ - macroscopic scattering cross section from group g' to g $[cm^{-1}]$;

k_{eff} - reactor multiplication factor;

χ_g - fission spectrum;

$\nu\Sigma_{fg'}(r)$ - number of neutrons emitted per fission times macroscopic fission cross section $[cm^{-1}]$;

$D_g(r)$ - diffusion coefficient $[cm]$.

In the steady-state reactor calculations, the system of equations (2.1-2.2) is a generalized eigenvalue problem and we are looking for the first eigenvalue k_{eff} and the first eigenvector $\varphi_g(x,y,z)$, $g=1,...,G$. It is convenient to convert the system (2.1-2.2) into the matrix form as

$$\sum_{u=x,y,z}\frac{\partial}{\partial u}\mathbf{j}_u(x,y,z)+\Sigma_r(x,y,z)\varphi(x,y,z)=\Sigma_s(x,y,z)\varphi(x,y,z) + \frac{1}{k_{eff}}\chi\nu\Sigma_f^T(x,y,z)\varphi(x,y,z) \quad (2.3)$$

$$\mathbf{j}_u(x,y,z)=-D(x,y,z)\frac{\partial}{\partial u}\varphi(x,y,z), u=x,y,z, \quad (2.4)$$

where

$\mathbf{j}_u(x,y,z)=col\{j_{1u}(x,y,z),...,j_{Gu}(x,y,z)\}, u=x,y,z$;

$\varphi(x,y,z)=col\{\varphi_1(x,y,z),...,\varphi_G(x,y,z)\}$;

$\Sigma_r(x,y,z)=\mathrm{diag}\{\Sigma_{r1}(x,y,z),\ldots,\Sigma_{rG}(x,y,z)\}$;

$\chi=\mathrm{col}\{\chi_1,\ldots,\chi_G\}$;

$\nu\Sigma_f(x,y,z)\mathrm{col}\{\nu\Sigma_{f1}(x,y,z),\ldots,\nu\Sigma_{fG}(x,y,z)\}$;

$\mathbf{D}(x,y,z)=\mathrm{diag}\{D_1(x,y,z),\ldots,D_G(x,y,z)\}$;

$$\Sigma_s(r) = \begin{bmatrix} 0 & \Sigma_{s12}(x,y,z) & \cdots & \Sigma_{s1G}(x,y,z) \\ \Sigma_{s21}(x,y,z) & 0 & \cdots & \Sigma_{s2G}(x,y,z) \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{sG1}(x,y,z) & \Sigma_{sG2}(x,y,z) & \cdots & 0 \end{bmatrix},$$

index T means a transpose operation.

The system of equations (2.3-2.4) should be completed by boundary conditions. We assume that the reactor domain is enclosed by the boundary normal to either x, y or z directions as illustrated in Fig.2.1 for two-dimensional case.

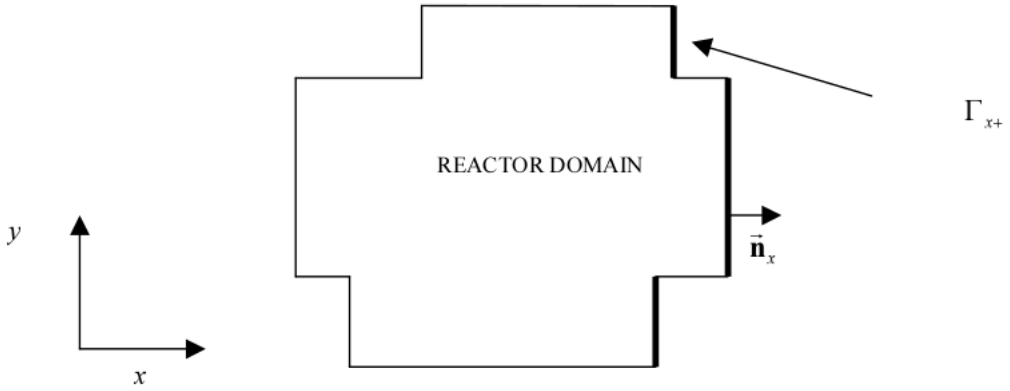


Figure 2.1: Example of the reactor domain and boundary in two-dimensional case.

Let us consider as an example a part of the boundary Γ_{x+} with the normal in plus-x-direction as shown in Fig.2.1. We use the following form of the boundary conditions

$$f\varphi(x,y,z) - \vec{\mathbf{n}}_x \cdot \vec{\mathbf{i}}_x \mathbf{B}\mathbf{j}_x(x,y,z) = 0, \quad (x,y,z) \in \Gamma_{x+}, \tag{2.5}$$

where

$\vec{\mathbf{n}}_x$ is the unit vector normal to the boundary Γ_{x+} and directed outwards;

$\vec{\mathbf{i}}_x$ is the unit vector in x-direction;

scalar f and $G \times G$ matrix $\mathbf{B}$ are constants determined by the certain form of

the boundary conditions.

In the reactor calculations the following boundary conditions can be considered. Zero neutron flux at the boundary defined as

$$\varphi(x,y,z)=0,(x,y,z)\in\Gamma_{x+}$$

is set up by

$$f=1;\mathbf{B}=\mathbf{0}.$$

Zero net neutron current boundary condition defined as

$$\mathbf{j}_x(x,y,z)=0,(x,y,z)\in\Gamma_{x+}$$

is given by

$$f=0;\mathbf{B}=\mathbf{I},$$

where $\mathbf{I}$ is the identity matrix.

The other two popular choices are defined in terms of the incoming and outcoming partial neutron currents. In diffusion theory, the partial currents satisfy the relations

$$2\left[\mathbf{j}_x^{\text{out}}(x,y,z)+\mathbf{j}_x^{\text{in}}(x,y,z)\right]=\varphi(x,y,z),$$

$$\mathbf{j}_x^{\text{out}}(x,y,z)-\mathbf{j}_x^{\text{in}}(x,y,z)=\pm\mathbf{j}_x(x,y,z),\quad(x,y,z)\in\Gamma_{x\pm},$$

where $+$ stands for the plus- x -directed part of the boundary Γ_{x+}, - for the minus- x -directed part of the boundary Γ_{x-}.

Zero incoming current boundary condition defined as

$$\mathbf{j}_x^{\text{in}}(x,y,z)=0,\quad(x,y,z)\in\Gamma_{x\pm}$$

is set up by

$$f=1;\mathbf{B}=2\mathbf{I}.$$

Albedo boundary condition defined as

$$\mathbf{j}_x^{\text{in}}(x,y,z)=\alpha\mathbf{j}_x^{\text{out}}(x,y,z),\quad(x,y,z)\in\Gamma_{x\pm},$$

is given by

$$f = 1; \mathbf{B} = 2\left[\mathbf{I} + \alpha\right]\left[\mathbf{I} - \alpha\right]^{-1},$$

where α is $G \times G$ albedo matrix.

2.2 Mesh-Centered Finite-Difference Method

To solve the neutron diffusion equations on computer, the continuous form (2.3-2.4) should be converted into a discrete representation. The reactor domain is subdivided into parallelepipeds called nodes with the grid indices x_l, y_m and z_n. A node (l, m, n) is defined as $x \in [x_l, x_{l+1}], y \in [y_m, y_{m+1}], z \in [z_n, z_{n+1}]$. It is convenient to move an origin of the local coordinates into the center of the node and define this node as a node k. The node widths are

$$\Delta x_k = x_{l+1} - x_l, \quad \Delta y_k = y_{m+1} - y_m, \quad \Delta z_k = z_{n+1} - z_n$$

and, in the local coordinates, the k-th node is

$$(x, y, z): \quad x \in \left[-\frac{\Delta x_k}{2}, \frac{\Delta x_k}{2}\right], \quad y \in \left[-\frac{\Delta y_k}{2}, \frac{\Delta y_k}{2}\right], \quad z \in \left[-\frac{\Delta z_k}{2}, \frac{\Delta z_k}{2}\right]$$

A node volume is defined as a product of the mesh spacing

$$V^k \equiv \Delta x_k \, \Delta y_k \, \Delta z_k.$$

In the following development, we omit a node index on the mesh spacing. We use $x+$ and $x-$ indices to denote the plus- x -directed (right) face of the node and the minus- x -directed (left) face of the node. The similar notations are applied for y- and z-directed faces. Material properties are assumed constant over the node and continuity of the flux and normal component of the current are imposed at internal faces of the node. Integrating (2.3) over the node volume we obtain the nodal balance equation

$$\sum_{u=x,y,z} F_u^k [\mathbf{J}_{u+}^k - \mathbf{J}_{u-}^k] + \Sigma_r^k \overline{\Phi}^k V^k = \Sigma_s^k \overline{\Phi}^k V^k + \frac{1}{k_{eff}} \chi (\nu \Sigma_f^k)^T \overline{\Phi}^k V^k, \tag{2.6}$$

where

$\overline{\Phi}^k \equiv \frac{1}{V^k}\int_{-\Delta x/2}^{\Delta x/2} dx \int_{-\Delta y/2}^{\Delta y/2} dy \int_{-\Delta z/2}^{\Delta z/2} dz\, \varphi(x,y,z)$- node-averaged neutron flux;

$\mathbf{J}^k_{x\pm}$ are the x-component of the net current averaged over the x-directed face of the node as

$$\mathbf{J}^k_{x\pm} \equiv \frac{1}{F^k_x}\int_{-\Delta z/2}^{\Delta z/2} dz \int_{-\Delta y/2}^{\Delta y/2} dy\, \mathbf{j}_x(x,y,z)\Big|_{x\to\pm\Delta x/2};$$

$F^k_x \equiv \Delta y \Delta z$ is the area of the x-directed face of the node.

Integrating the Fick's law (2.4) over the x-directed face of the node we get

$$\mathbf{J}^k_{x\pm} = -\mathbf{D}^k \frac{d}{dx}\Phi^k_x(x)\Big|_{x\to\pm\Delta x/2}, \tag{2.7}$$

where

$\Phi^k_x(x) \equiv \frac{1}{F^k_x}\int_{-\Delta z/2}^{\Delta z/2} dz \int_{-\Delta y/2}^{\Delta y/2} dy\, \varphi(x,y,z)$ is the face-averaged (transverse-integrated) neutron flux.

Neutron balance equations (2.6-2.7) are formally exact, but to solve them we have to add the relations between the face-averaged and the node-averaged fluxes. Let us consider, for example, the plus- x-directed (right) face of the node $x+$. In the finite-difference method, the derivative in Eq. (2.7) is replaced by the simple finite-difference as

$$\mathbf{J}^k_{x+} = -\hat{\mathbf{D}}^k_x\left[\Phi^k_{x+} - \overline{\Phi}^k\right], \tag{2.8}$$

where $\hat{\mathbf{D}}^k_x = 2\mathbf{D}^k/\Delta x$ is the nondimensional diffusion coefficient.

Let us assume that the neighbouring node $(l+1,m,n)$ has the index $k+1$. Using the similar expression for the current at the left face of the node $k+1$ and continuity of the face-averaged flux and current we express the face-averaged flux in terms of the node-averaged fluxes as

$$\Phi^k_{x+} = \Phi^{k+1}_{x-} = \left[\widehat{\mathbf{D}}^k_x + \widehat{\mathbf{D}}^{k+1}_x\right]^{-1}\left(\widehat{\mathbf{D}}^{k+1}_x\,\overline{\Phi}^{k+1} + \widehat{\mathbf{D}}^k_x\,\overline{\Phi}^k\right),$$

Substituting the expression for the face-averaged flux into (2.8) we express the

face-averaged current as

$$\mathbf{J}_{x+}^{k} = \mathbf{J}_{x-}^{k+1} = -\hat{\mathbf{D}}_{x}^{k+1}\left[\hat{\mathbf{D}}_{x}^{k+1} + \hat{\mathbf{D}}_{x}^{k}\right]^{-1}\hat{\mathbf{D}}_{x}^{k}\left(\bar{\Phi}^{k+1} - \bar{\Phi}^{k}\right)$$

In the case of node face on the boundary we use the boundary condition (2.5) instead of the flux and current continuity. The face-averaged neutron flux in the case of the boundary face is given as

$$\Phi_{x\pm}^{k} = \left[f_{x\pm}^{k} + \mathbf{B}_{x\pm}^{k}\hat{\mathbf{D}}_{x}^{k}\right]^{-1}\mathbf{B}_{x\pm}^{k}\hat{\mathbf{D}}_{x}^{k}\overline{\Phi}^{k},$$

where $f_{x\pm}^{k}$ and $\mathbf{B}_{x\pm}^{k}$ are the constants of the boundary conditions at the $x\pm$ boundary face of the node. Then, the face-averaged current at the $x\pm$ boundary face of the node is expressed as

$$\mathbf{J}_{x\pm}^{k} = \pm\hat{\mathbf{D}}_{x}^{k}\left[f_{x\pm}^{k} + \mathbf{B}_{x\pm}^{k}\hat{\mathbf{D}}_{x}^{k}\right]^{-1} f_{x\pm}^{k}\,\bar{\Phi}^{k}.$$

Substituting the above expressions for the face-averaged current into the neutron balance equation (2.6) we obtain the mesh-centered finite-difference form of the neutron diffusion equation as

$$\sum_{u=x,y,z} F_{u}^{k}\left\{-\tilde{\mathbf{D}}_{u}^{k}\overline{\Phi}^{k+1} + [\tilde{\mathbf{D}}_{u}^{k} + \tilde{\mathbf{D}}_{u}^{k-1}]\overline{\Phi}^{k} - \tilde{\mathbf{D}}_{u}^{k-1}\overline{\Phi}^{k-1}\right\} + \Sigma_{r}^{k}\overline{\Phi}^{k}V^{k} = \Sigma_{s}^{k}\bar{\Phi}^{k}V^{k} + \frac{1}{k_{eff}}\chi(\nu\Sigma_{f}^{k})^{T}\overline{\Phi}^{k}V^{k}, \quad (2.9)$$

where

$k+1$ $(k-1)$ is the index of the neighboring node in the positive (negative) u-direction;

$\tilde{\mathbf{D}}_{u}^{k} = \hat{\mathbf{D}}_{u}^{k+1}\left[\hat{\mathbf{D}}_{u}^{k+1} + \hat{\mathbf{D}}_{u}^{k}\right]^{-1}\hat{\mathbf{D}}_{u}^{k}$ is the finite-difference coupling coefficient for internal interfaces;

$\tilde{\mathbf{D}}_{u}^{k} = \hat{\mathbf{D}}_{u}^{k}\left[f_{x\pm}^{k} + \mathbf{B}_{x\pm}^{k}\hat{\mathbf{D}}_{x}^{k}\right]^{-1} f_{x\pm}^{k}$ is the finite-difference coupling coefficient for boundary interfaces.

The finite-difference Eq. (2.9) is ready for solving on a computer. There is however a problem of accuracy. Accurate finite-difference calculation requires that the mesh spacing should be an order of diffusion length, which is on the order of a centimetre in the case of light water reactor. This requirement results in hundreds of thousands unknowns making the calculations too time-consuming. Next sections will show a way to improve an accuracy of the finite-difference method, without significant modifications of the Eq. (2.9).

2.3 Nonlinear Iteration Procedure

Kord Smith proposed a simple way to improve accuracy of the finite-difference method without significant modifications (Smith, 1984). In his approach, called "nonlinear iteration procedure", the coarse-mesh finite-difference method is forced to match the results of the other discretization scheme, for example a nodal method. The approach lacking a theoretical basis but proved to be very efficient. It is used in several neutron kinetics codes with different nodal methods: in the NESTLE (Al-Chalabi et al., 1993), SPANDEX (Aviles, 1993) and PARCS (Joo et al., 1996) codes with nodal expansion method (NEM) and in the SIMULATE-3K (Borkowski et al., 1996) code with the semi-analytical nodal method. There are several variants different in the way of correction of the finite-difference equations. We adopted an approach used in the NESTLE code (Turinsky et al., 1994). The surface-averaged neutron current is expressed as

$$\mathbf{J}_{u+}^{k} = -\tilde{\mathbf{D}}_{u}^{k}\left(\overline{\Phi}^{k+1} - \overline{\Phi}^{k}\right) - \tilde{\mathbf{D}}_{u}^{k}(nod)\left(\overline{\Phi}^{k+1} + \overline{\Phi}^{k}\right), \tag{2.10}$$

where

- $k+1$ is the index of the neighboring node in the positive u-direction;
- $\tilde{\mathbf{D}}_{u}^{k}$ is the matrix of finite-difference coupling coefficients derived in the previous section;
- $\tilde{\mathbf{D}}_{u}^{k}(nod)$ is a matrix of nodal coupling coefficients, which will be defined below.

Substituting this expression for the neutron current into the neutron balance equation (2.6), we obtain a new coarse-mesh finite-difference form of the neutron diffusion equations:

$$\sum_{u=x,y,z} F_u^k \Big\{ -\left[\tilde{\mathbf{D}}_u^k + \tilde{\mathbf{D}}_u^k(nod)\right]\overline{\Phi}^{k+1} + \left[\tilde{\mathbf{D}}_u^k - \tilde{\mathbf{D}}_u^k(nod) + \tilde{\mathbf{D}}_u^{k-1} + \tilde{\mathbf{D}}_u^{k-1}(nod)\right]\overline{\Phi}^k - \\ - \left[\tilde{\mathbf{D}}_u^{k-1} - \tilde{\mathbf{D}}_u^{k-1}(nod)\right]\overline{\Phi}^{k-1} \Big\} + \Sigma_r^k \overline{\Phi}^k V^k = \Sigma_s^k \overline{\Phi}^k V^k + \frac{1}{k_{eff}} \chi (\nu\Sigma_f^k)^T \overline{\Phi}^k V^k \tag{2.11}$$

Eq. (2.11) is called the coarse-mesh finite-difference (CMFD) equation to distinguish it from the standard form of the finite-difference equation (2.9). The nonlinear iterations start with the nodal coupling coefficients $\tilde{\mathbf{D}}_u^k(nod)$ set up to 0. After several iterations solving CMFD equations (2.11) we get the new approximation for the node-averaged neutron fluxes and the eigenvalue. Using the equations of the nodal method we compute the values of the surface-average neutron current. In the CMFD method, the surface-averaged neutron current is determined by Eq. (2.10). Requiring that the CMFD method reproduce the same values of the surface-averaged neutron current we compute the nodal coupling coefficients as

$$\left(\tilde{D}_u^k(nod)\right)_{gg} = \frac{-J_{gu^+}^k - \tilde{D}_{gu}^k\left(\overline{\Phi}_g^{k+1} - \overline{\Phi}_g^k\right)}{\overline{\Phi}_g^{k+1} + \overline{\Phi}_g^k}; \quad \left(\tilde{D}_u^k(\mathrm{nod})\right)_{gg'} = 0 \quad \text{if} \quad g' \neq g \tag{2.12}$$

The iterations are performed till convergence, when the nodal coupling coefficients do not change anymore. The total solution procedure is decoupled into two processes: iterative solution of the CMFD equations (2.11), where we compute the node-averaged neutron fluxes and eigenvalue; and solution of the nodal equations to compute the face-averaged neutron currents and nodal coupling coefficients by Eq. (2.12).

Derivation of the nodal equations used to compute the face-averaged neutron current is given in the following sections.

2.4 Transverse-Integration Procedure

Nodal methods used in the SKETCH-N code are based on the transverse-integration procedure proposed by H. Finnemann (1975). In this approach, three-dimensional neutron diffusion equations (2.3-2.4) are integrated in two directions transverse to the direction of the interest. In the case of x direction, the transverse integration operator is defined as

$$\frac{1}{F_x^k}\int_{-\Delta y/2}^{\Delta y/2} dy \int_{-\Delta z/2}^{\Delta z/2} dz.$$

As a result we obtain the 1D diffusion equations:

$$\frac{d}{dx}\mathbf{J}_x^k(x) + \Sigma_r^k \Phi_x^k(x) = \Sigma_s^k \Phi_x^k(x) + \frac{1}{k_{eff}}\chi(\nu\Sigma_f^k)^T \Phi_x^k(x) - \mathbf{S}_x^k(x), \tag{2.13}$$

$$\mathbf{J}_x^k(x) = -\mathbf{D}^k \frac{d}{dx}\Phi_x^k(x), \quad x \in [x_l, x_{l+1}], \tag{2.14}$$

where

$\Phi_x^k(x) \equiv \frac{1}{F_x^k}\int_{-\Delta y/2}^{\Delta y/2} dy \int_{-\Delta z/2}^{\Delta z/2} dz\, \varphi(x,y,z)$ is the transverse-integrated neutron flux;

$\mathbf{J}_x^k(x) \equiv \frac{1}{F_x^k}\int_{-\Delta y/2}^{\Delta y/2} dy \int_{-\Delta z/2}^{\Delta z/2} dz\, \mathbf{j}_x(x,y,z)$ is the transverse-integrated neutron current;

$\mathbf{S}_x^k(x) \equiv \frac{1}{\Delta y}\mathbf{L}_{xy}^k(x) + \frac{1}{\Delta z}\mathbf{L}_{xz}^k(x)$ is the transverse leakage source;

$\mathbf{L}_{xy}^k(x) \equiv \frac{1}{\Delta z}\int_{-\Delta z/2}^{\Delta z/2} dz\, [\mathbf{j}_y(x, \Delta y/2, z) - \mathbf{j}_y(x, -\Delta y/2, z)]$;

$\mathbf{L}_{xz}^k(x) \equiv \frac{1}{\Delta y}\int_{-\Delta y/2}^{\Delta y/2} dy\, [\mathbf{j}_z(x, y, \Delta z/2) - \mathbf{j}_z(x, y, -\Delta z/2)]$.

Applying the same procedure for the y and z directions we get a system of three quasi-one-dimensional equations coupled by their right hand sides through the transverse leakages.

Before proceeding further it is convenient to map the interval $[-\Delta x/2, \Delta x/2]$ onto $[-1, +1]$ and rewrite equations (2.13-2.14) as

$$-\frac{d^2}{d\xi^2}\Phi_x^k(\xi) + \left(\mathbf{B}^2\right)_x^k \Phi_x^k(\xi) = -\hat{\mathbf{S}}_x^k(\xi), \tag{2.15}$$

$$\mathbf{J}_x^k(\xi) = -\hat{\mathbf{D}}_x^k \frac{d}{d\xi}\Phi_x^k(\xi), \quad \xi \in [-1,1], \tag{2.16}$$

where

$\xi = \frac{2x}{\Delta x_k}$;

$(\mathbf{B}^2)_x^k = \frac{(\Delta x)^2}{4}\left[\mathbf{D}^k\right]^{-1}\left\{\Sigma_r^k - \Sigma_s^k - \frac{1}{k_{eff}}\chi(\nu\Sigma_f^k)^T\right\}$ - nondimensional buckling matrix;

$\hat{\mathbf{S}}_x^k(\xi) = \frac{(\Delta x)^2}{4}\left[\mathbf{D}^k\right]^{-1}\mathbf{S}_x^k(x)$.

Assuming that the transverse leakage $\hat{\mathbf{S}}_x^k(\xi)$ is known, Eqs. (2.15-2.16) can be easily solved by various methods.

2.5 Transverse Leakage Approximations

To solve the transverse-integrated equations (2.15–2.16), we need additional information on the transverse leakage. The first approximation can be derived assuming that the neutron flux inside the node is separable as:

$$\varphi^k(x,y,z) = \Phi_x^k(x) + \Phi_y^k(y) + \Phi_z^k(z) - 2\overline{\Phi}^k$$

Then, using the definition of the transverse leakage, we have that the transverse leakage is constant inside the node:

$$\mathbf{S}_x^k(x) = \overline{\mathbf{S}}_x^k = \frac{1}{\Delta y}\overline{\mathbf{L}}_{xy}^k + \frac{1}{\Delta z}\overline{\mathbf{L}}_{xz}^k = \frac{1}{\Delta y}\left[\mathbf{J}_{y+}^k - \mathbf{J}_{y-}^k\right] + \frac{1}{\Delta z}\left[\mathbf{J}_{z+}^k - \mathbf{J}_{z-}^k\right]$$

This approach is called a *flat leakage approximation* (Shober et al., 1977). The flat leakage approximation leads to relatively large errors in the case of realistic LWR problems, and an improved variant, called *quadratic leakage approximation* (Finnemann, 1975), has been proposed. The transverse leakage inside the node is approximated by a quadratic polynomial as:

$$\mathbf{S}_x^k(x) = \overline{\mathbf{S}}_x^k + \sum_{i=1}^{2} \mathbf{s}_{xi}^k P_i(2x/\Delta x),$$

where $P_i(\xi)$ are some polynomials, which in our case are Legendre polynomials, i.e., $P_1(\xi) = \xi$ and $P_2(\xi) = \frac{1}{2}(3\xi^2 - 1), \quad \xi = 2x/\Delta x$.

There are several ways to compute the 1st and 2nd expansion coefficients (Lawrence, 1986). We use the traditional approach extending the interpolating polynomial over the node k and its two neighboring nodes $k-1$ and $k+1$. Then we require that the polynomial reproduces the same average transverse leakages also for the nodes $k-1$ and $k+1$. Thus, we have two additional equations to compute the 1st and 2nd expansion coefficients. The explicit expressions are given as:

$$\mathbf{s}_{x1}^k = \left[g_x^k\right]^{-1} \left[(\overline{\mathbf{S}}_x^{k+1} - \overline{\mathbf{S}}_x^k)(1+\delta_x^{k-1})(1+2\delta_x^{k-1}) + (\overline{\mathbf{S}}_x^k - \overline{\mathbf{S}}_x^{k-1})(1+\delta_x^{k+1})(1+2\delta_x^{k+1})\right];$$

$$\mathbf{s}_{x2}^k = \left[g_x^k\right]^{-1} \left[\left(\overline{\mathbf{S}}_x^{k+1} - \overline{\mathbf{S}}_x^k\right)\left(1+\delta_x^{k-1}\right) + \left(\overline{\mathbf{S}}_x^{k-1} - \overline{\mathbf{S}}_x^k\right)\left(1+\delta_x^{k+1}\right)\right];$$

where the normalization factor is:

$$g_x^k = 2\left(1+\delta_x^{k-1}\right)\left(1+\delta_x^{k+1}\right)\left(1+\delta_x^{k+1}+\delta_x^{k-1}\right);$$

and the mesh ratio parameters are defined as:

$$\delta_x^{k-1} = h_x^{k-1}/h_x^k; \quad \delta_x^{k+1} = h_x^{k+1}/h_x^k.$$

In the case where the node does not have a neighbor, we use the boundary condition (2.5) instead of the requirement to reproduce the average over the node transverse leakage.

-

2.6 Polynomial Nodal Method (PNM)

In this section, we present the Polynomial Nodal Method (PNM) to solve Eqs. (2.15–2.16). The transverse-integrated neutron flux is expanded into Legendre polynomials up to the 4^{th} order as:

$$\Phi_x^k(\xi) = \overline{\Phi}^k + \sum_{i=1}^{4} \mathbf{a}_{xi}^k P_i(\xi), \tag{2.17}$$

where:
$\overline{\Phi}^k$ is the node-averaged neutron flux;

$\mathbf{a}_{xi}^k = \text{col}\{a_{1xi}^k, \ldots, a_{Gxi}^k\}$ is a vector of the unknown flux expansion coefficients;

$P_i(\xi)$ are Legendre polynomials, defined as:

$$\begin{aligned}
P_0(\xi) &= 1; \\
P_1(\xi) &= \xi; \\
P_2(\xi) &= \frac{1}{2}(3\xi^2 - 1); \\
P_3(\xi) &= \frac{1}{2}(5\xi^3 - 3\xi); \\
P_4(\xi) &= \frac{1}{8}(35\xi^4 - 30\xi^2 + 3).
\end{aligned}$$

Legendre polynomials are orthogonal on the interval $[-1, 1]$, satisfying:

$$\int_{-1}^{+1} P_i(\xi) P_j(\xi)\, d\xi = 0 \quad \text{if } i \neq j$$

and

$$\int_{-1}^{+1} P_i^2(\xi)\, d\xi = N_i = \frac{2}{2i+1}.$$

Orthogonality of the Legendre polynomials results in a simple set of the final nodal equations. In the nonlinear iteration procedure, the node-averaged neutron flux and eigenvalue are assumed known from the results of the CMFD iterations.

We need to compute the face-averaged currents for each node interface. Using the flux expansion (2.17) we expressed the face-averaged current in terms of the flux expansion coefficients as

$$\mathbf{J}^k_{x+} = \mathbf{J}^k_x(+1) = -\hat{\mathbf{D}}^k_x \left\{ \mathbf{a}^k_{x1} + 3\,\mathbf{a}^k_{x2} + 6\,\mathbf{a}^k_{x3} + 10\,\mathbf{a}^k_{x4} \right\} \tag{2.18}$$

The unknown expansion coefficients are computed by considering a two-node problem for each node interface, as illustrated in Fig.2.2.

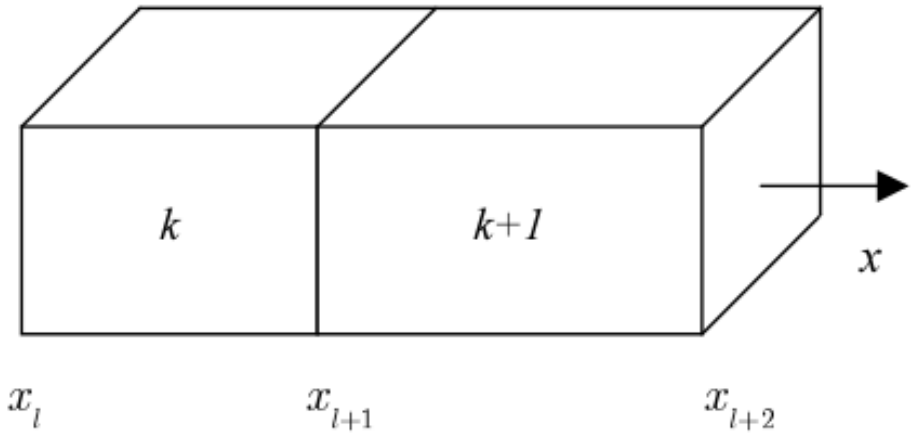


Figure 2.2: Two-node problem $(k, k+1)$

For this two-node problem we have four unknown expansion coefficients per node per energy group or eight expansion coefficients per group per two-node problem. Thus, eight equations are required to compute them. The equations are

1. Nodal balance equations for the nodes k and k+1 (2G equations)
2. Two moment-weighting equations for the nodes k and k+1 (4G equations)
3. Continuity of the face-averaged flux at an internal face of the two-node problem (G equations)
4. Continuity of the face-averaged current at an internal face of the two-node problem (G equations).

In the case of the face at the boundary, we define a one-node problem for the boundary node. In this case, we have 4G unknowns. The flux and current continuity equations are replaced by the boundary condition (2.5) and we have 4G nodal equations for 4G unknowns. Substituting expansion (2.17) into Eq. (2.15) and integrating over the interval [−1, +1] yields the *nodal balance equations*:

$$-\left\{3\mathbf{a}_{x2}^{k}+10\mathbf{a}_{x4}^{k}\right\}+\left(\mathbf{B}^{2}\right)_{x}^{k}\overline{\Phi}^{k}=-\hat{\mathbf{s}}_{x0}^{k}, \tag{2.19}$$

where

$$\hat{\mathbf{s}}_{x0}^{k}=\frac{1}{N_0}\int_{-1}^{+1}\hat{\mathbf{S}}_{x}^{k}(\xi)\,d\xi=\frac{(\Delta x)^{2}}{4}\left[\mathbf{D}^{k}\right]^{-1}\overline{\mathbf{S}}_{x}^{k}.$$

Moment-weighting equations are obtained by moment-weighting with $P_1(\xi)$ and $P_2(\xi)$ yielding 1st-order moment-weighting equations (for nodes k and $k+1$):

$$-15\mathbf{a}_{x3}^{k}+\left(\mathbf{B}^{2}\right)_{x}^{k}\mathbf{a}_{x1}^{k}=-\hat{\mathbf{s}}_{x1}^{k}, \tag{2.20}$$

where

$$\hat{\mathbf{s}}_{x1}^{k}=\frac{1}{N_1}\int_{-1}^{+1}P_1(\xi)\hat{\mathbf{S}}_{x}^{k}(\xi)\,d\xi=\frac{(\Delta x)^{2}}{4}\left[\mathbf{D}^{k}\right]^{-1}\mathbf{s}_{x1}^{k}.$$

2nd-order moment-weighting equations (for nodes k and $k+1$):

$$-35\mathbf{a}_{x4}^{k}+\left(\mathbf{B}^{2}\right)_{x}^{k}\mathbf{a}_{x2}^{k}=-\hat{\mathbf{s}}_{x2}^{k}, \tag{2.21}$$

where

$$\hat{\mathbf{s}}_{x2}^{k}=\frac{1}{N_2}\int_{-1}^{+1}P_2(\xi)\hat{\mathbf{S}}_{x}^{k}(\xi)\,d\xi=\frac{(\Delta x)^{2}}{4}\left[\mathbf{D}^{k}\right]^{-1}\mathbf{s}_{x2}^{k}.$$

Continuity of the transverse-integrated neutron flux is written as

$$\overline{\Phi}^{k}+\mathbf{a}_{x1}^{k}+\mathbf{a}_{x2}^{k}+\mathbf{a}_{x3}^{k}+\mathbf{a}_{x4}^{k}=\overline{\Phi}^{k+1}-\mathbf{a}_{x1}^{k+1}+\mathbf{a}_{x2}^{k+1}-\mathbf{a}_{x3}^{k+1}+\mathbf{a}_{x4}^{k+1} \tag{2.22}$$

Continuity of the transverse-integrated neutron current is written as

$$-\hat{\mathbf{D}}_x^k \left\{\mathbf{a}_{x1}^k + 3\mathbf{a}_{x2}^k + 6\mathbf{a}_{x3}^k + 10\mathbf{a}_{x4}^k\right\} = -\hat{\mathbf{D}}_x^{k+1} \left\{\mathbf{a}_{x1}^{k+1} - 3\mathbf{a}_{x2}^{k+1} + 6\mathbf{a}_{x3}^{k+1} - 10\mathbf{a}_{x4}^{k+1}\right\} \tag{2.23}$$

The obtained equations are decoupled. The even flux expansion coefficients of the node k do not depend on the odd flux expansion coefficients of the node k and any expansion coefficients of the node $k+1$. Furthermore, using 1$^{\text{st}}$-order moment weighting equation (2.20), the 3$^{\text{rd}}$ expansion coefficients are expressed in terms of the 1$^{\text{st}}$ expansion coefficients as

$$\mathbf{a}_{x3}^k = \frac{1}{15}\left\{\left(\mathbf{B}^2\right)_x^k \mathbf{a}_{x1}^k + \hat{\mathbf{s}}_{x1}^k\right\} \tag{2.24}$$

Using 2$^{\text{nd}}$-order moment weighting equation (2.21), the 4$^{\text{th}}$ expansion coefficients are expressed in terms of the 2$^{\text{nd}}$ expansion coefficients as

$$\mathbf{a}_{x4}^k = \frac{1}{35}\left\{\left(\mathbf{B}^2\right)_x^k \mathbf{a}_{x2}^k + \hat{\mathbf{s}}_{x2}^k\right\} \tag{2.25}$$

Substituting Eq. (2.25) into the neutron balance equation (2.19) we obtain a system of G equations for the 2$^{\text{nd}}$ expansion coefficients

$$\left\{3\mathbf{I} + \frac{2}{7}\left(\mathbf{B}^2\right)_x^k\right\}\mathbf{a}_{x2}^k = \left(\mathbf{B}^2\right)_x^k \overline{\Phi}^k + \left\{\hat{\mathbf{s}}_{x0}^k - \frac{2}{7}\hat{\mathbf{s}}_{x2}^k\right\}, \tag{2.26}$$

where $\mathbf{I}$ is the identity matrix.

Substituting Eq. (2.24) into the neutron flux and neutron current continuity equations (2.22-2.23) we obtain $2G$ equations for the 1$^{\text{st}}$ expansion coefficients:

Flux continuity:

$$\left\{\mathbf{I} + \frac{1}{15}\left(\mathbf{B}^2\right)_x^{k+1}\right\}\mathbf{a}_{x1}^{k+1} + \left\{\mathbf{I} + \frac{1}{15}\left(\mathbf{B}^2\right)_x^k\right\}\mathbf{a}_{x1}^k = \left\{\overline{\Phi}^{k+1} + \mathbf{a}_{x2}^{k+1} + \mathbf{a}_{x4}^{k+1}\right\} - \left\{\overline{\Phi}^k + \mathbf{a}_{x2}^k + \mathbf{a}_{x4}^k\right\} - \frac{1}{15}\left\{\hat{\mathbf{s}}_{x1}^k + \hat{\mathbf{s}}_{x1}^{k+1}\right\} \tag{2.27}$$

Current continuity:

$$\hat{\mathbf{D}}_x^{k+1}\left\{\mathbf{I}+\frac{2}{5}\left(\mathbf{B}^2\right)_x^{k+1}\right\}\mathbf{a}_{x1}^{k+1}-\hat{\mathbf{D}}_x^{k}\left\{\mathbf{I}+\frac{2}{5}\left(\mathbf{B}^2\right)_x^{k}\right\}\mathbf{a}_{x1}^{k}= \\ \left\{\hat{\mathbf{D}}_x^{k+1}\left\{3\mathbf{a}_{x2}^{k+1}+10\mathbf{a}_{x4}^{k+1}-\frac{2}{5}\hat{\mathbf{s}}_{x1}^{k+1}\right\}+\hat{\mathbf{D}}_x^{k}\left\{3\mathbf{a}_{x2}^{k}+10\mathbf{a}_{x4}^{k}+\frac{2}{5}\hat{\mathbf{s}}_{x1}^{k}\right\}\right\} \quad (2.28)$$

As a result, the initial system of $8G$ nodal equations is reduced to G equations (2.26) with respect to the 2nd expansion coefficients of the node $k+1$ and $2G$ equations (2.27–2.28) with respect to the 1st expansion coefficients of the nodes $k+1$ and k. The even expansion coefficients of the node k are known from the solution of the previous two-node problem $(k-1,k)$. When the neutron flux expansion coefficients are computed, the face-averaged nodal current $\mathbf{J}_{x+}^k$ is calculated using Eq. (2.18).

It is worth noting that Legendre polynomials $P_i(u)$ and the NEM basic functions (Finnemann et al., 1977) adjusted to the interval $[-1,+1]$ satisfy the following relation:

$$\begin{pmatrix} P_0^{\mathrm{NEM}}(u) \\ P_1^{\mathrm{NEM}}(u) \\ P_2^{\mathrm{NEM}}(u) \\ P_3^{\mathrm{NEM}}(u) \\ P_4^{\mathrm{NEM}}(u) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 \\ 0 & -1/20 & 0 & 1/20 & 0 \\ 0 & 0 & -1/70 & 0 & 1/70 \end{pmatrix} \begin{pmatrix} P_0(u) \\ P_1(u) \\ P_2(u) \\ P_3(u) \\ P_4(u) \end{pmatrix}.$$

NEM basic functions span the same subspace as Legendre polynomials. If the moment weighting procedure is used in NEM, PNM and NEM become equivalent. In our opinion, the nodal equations of the PNM method are simpler than those of NEM.

2.7 Semi-Analytic Nodal Method (SANM)

In the semi-analytic nodal method (SANM), the 3rd and 4th Legendre polynomials are replaced by hyperbolic sine and cosine, which are general solutions of the following homogeneous problem:

$$-\frac{d^2}{d\xi^2}\Phi_{gx}^k(\xi) + \left(\alpha_{gx}^k\right)^2 \Phi_g^k(\xi) = 0,$$

where

$$\alpha_{gx}^k = \frac{\Delta x}{2}\sqrt{\left[D_g^k\right]^{-1}\Sigma_R^k}.$$

The sine and cosine are modified to preserve orthogonality of the basic functions and normalized to unity at the right end of the interval $[-1,+1]$, following the mathematical analytic nodal method (MANM) developed by J. P. Hemart (1988). The expressions for the 3rd and 4th basic functions are:

$$P_{g3}^{\mathrm{SANM}}(\xi) = \frac{\sinh(\alpha_{gx}^k\xi) - m_{gx1}^k(\sinh)P_1(\xi)}{\sinh(\alpha_{gx}^k) - m_{gx1}^k(\sinh)};$$

$$P_{g4}^{\mathrm{SANM}}(\xi) = \frac{\cosh(\alpha_{gx}^k\xi) - m_{gx0}^k(\cosh)P_0(\xi) - m_{gx2}^k(\cosh)P_2(\xi)}{\cosh(\alpha_{gx}^k) - m_{gx0}^k(\cosh) - m_{gx2}^k(\cosh)}$$

where

$$m_{gx1}^k(\sinh) = \frac{1}{N_1}\int_{-1}^{1}\sinh(\alpha_{gx}^k\xi)P_1(\xi)\,d\xi;$$

$$m_{gxi}^k(\cosh) = \frac{1}{N_i}\int_{-1}^{1}\cosh(\alpha_{gx}^k\xi)P_i(\xi)\,d\xi, \text{ for } i=0,2; \quad N_i = \frac{2}{2i+1}, \quad i=0,1,2.$$

As a result, the neutron flux is represented as:

$$\Phi_x^k(\xi) = \bar{\Phi}^k + \sum_{i=1}^{4}\mathbf{P}_i^{SANM}(\xi)\mathbf{a}_{xi}^k,$$

where:

$\mathbf{a}_{xi}^k = \mathrm{col}\{a_{1xi}^k,\ldots,a_{Gxi}^k\}$ is a vector of the unknown flux expansion coefficients;

$\mathbf{P}_i^{SANM}$ is the diagonal $G\times G$ matrix defined as:

$$\mathbf{P}_i^{SANM} = P_i\mathbf{I}, \quad \text{for } i=1,2;$$

$$\left(\mathbf{P}_i^{SANM}\right)_{gg} = P_{gi}^{\mathrm{SANM}}(\xi),$$

$$\left(\mathbf{P}_i^{SANM}\right)_{gg'} = 0, \quad \text{if } g'\neq g \quad \text{for } i=3,4.$$

The SANM equations are obtained similarly to the PNM method described in Section 2.6. The 3rd flux expansion coefficient is expressed as:

$$\mathbf{a}_{x3}^{k} = \mathbf{A}_{x}^{k}\left(\left(\mathbf{B}^{2}\right)_{x}^{k}\mathbf{a}_{x1}^{k} + \hat{\mathbf{s}}_{x1}^{k}\right), \tag{2.29}$$

where:

$$\left(A_{x}^{k}\right)_{gg} = \left(\frac{\sinh\left(\alpha_{gx}^{k}\right) - m_{gx1}^{k}(\sinh)}{\left(\alpha_{gx}^{k}\right)^{2} m_{gx1}^{k}(\sinh)}\right), \quad \left(A_{x}^{k}\right)_{gg'} = 0, \quad \text{if } g' \neq g.$$

The 4th expansion coefficient is expressed as:

$$\mathbf{a}_{x4}^{k} = \mathbf{T}_{x}^{k}\left(\left(\mathbf{B}^{2}\right)_{x}^{k}\mathbf{a}_{x2}^{k} + \hat{\mathbf{s}}_{x2}^{k}\right), \tag{2.30}$$

where:

$$\left(T_{x}^{k}\right)_{gg} = \frac{\cosh\left(\alpha_{gx}^{k}\right) - m_{gx0}^{k}(\cosh) - m_{gx2}^{k}(\cosh)}{\left(\alpha_{gx}^{k}\right)^{2} m_{gx2}^{k}(\cosh)}, \quad \left(T_{x}^{k}\right)_{gg'} = 0, \quad \text{if } g' \neq g.$$

Substituting Eq. (2.30) into the neutron balance equation and Eq. (2.29) into the flux and current continuity equations, we obtain G and $2G$ equations to solve the two-node problem:

Neutron balance equation:

$$\left\{3\mathbf{I} + \mathbf{E}_{x}^{k}\left(\mathbf{B}^{2}\right)_{x}^{k}\right\}\mathbf{a}_{x2}^{k} = \left(\mathbf{B}^{2}\right)_{x}^{k}\bar{\Phi}^{k} + \left\{\hat{\mathbf{s}}_{x0}^{k} - \mathbf{E}_{x}^{k}\hat{\mathbf{s}}_{x2}^{k}\right\}$$

where

$$\left(E_{x}^{k}\right)_{gg} = \left(\frac{m_{gx0}^{k}(\cosh)}{m_{gx2}^{k}(\cosh)} - \frac{3}{\left(\alpha_{gx}^{k}\right)^{2}}\right), \quad \left(E_{x}^{k}\right)_{gg'} = 0, \text{ if } g' \neq g$$

Flux continuity equation:

$$\left\{\mathbf{I} + \mathbf{A}_{x}^{k+1}\left(\mathbf{B}^{2}\right)_{x}^{k+1}\right\}\mathbf{a}_{x1}^{k+1} + \left\{\mathbf{I} + \mathbf{A}_{x}^{k}\left(\mathbf{B}^{2}\right)_{x}^{k}\right\}\mathbf{a}_{x1}^{k} = \left\{\bar{\Phi}^{k+1} + \mathbf{a}_{x2}^{k+1} + \mathbf{a}_{x4}^{k+1}\right\} - \left\{\bar{\Phi}^{k} + \mathbf{a}_{x2}^{k} + \mathbf{a}_{x4}^{k}\right\} - \left\{\mathbf{A}_{x}^{k}\hat{\mathbf{s}}_{x1}^{k} + \mathbf{A}_{x}^{k+1}\hat{\mathbf{s}}_{x1}^{k+1}\right\}$$

Current continuity equation:

$$\hat{\mathbf{D}}_x^{k+1}\left\{\mathbf{I}+\mathbf{F}_x^{k+1}\left(\mathbf{B}^2\right)_x^{k+1}\right\}\mathbf{a}_{x1}^{k+1}-\hat{\mathbf{D}}_x^{k}\left\{\mathbf{I}+\mathbf{F}_x^{k}\left(\mathbf{B}^2\right)_x^{k}\right\}\mathbf{a}_{x1}^{k}=$$
$$\hat{\mathbf{D}}_x^{k+1}\left\{3\mathbf{a}_{x2}^{k+1}+\mathbf{G}_x^{k+1}\mathbf{a}_{x4}^{k+1}-\mathbf{F}_x^{k+1}\hat{\mathbf{s}}_{x1}^{k+1}\right\}+\hat{\mathbf{D}}_x^{k}\left\{3\mathbf{a}_{x2}^{k}+\mathbf{G}_x^{k}\mathbf{a}_{x4}^{k}+\mathbf{F}_x^{k}\hat{\mathbf{s}}_{x1}^{k}\right\}$$

where

$$\left(F_x^k\right)_{gg}=\frac{\alpha_{gx}^k\cosh(\alpha_{gx}^k)-m_{gx1}^k(\sinh)}{\left(\alpha_{gx}^k\right)^2 m_{gx1}^k(\sinh)},\quad \left(F_x^k\right)_{gg'}=0,\ \text{if } g'\neq g;$$

$$\left(G_x^k\right)_{gg}=\frac{\alpha_{gx}^k\sinh(\alpha_{gx}^k)-3m_{gx2}^k(\cosh)}{\cosh(\alpha_{gx}^k)-m_{gx0}^k(\cosh)-m_{gx2}^k(\cosh)},\quad \left(G_x^k\right)_{gg'}=0,\ \text{if } g'\neq g.$$

When the nodal equations are solved, the face-averaged nodal current $\mathbf{J}_{x+}^k$ is computed as:

$$\mathbf{J}_{x+}^k=\mathbf{J}_x^k(+1)=-\hat{\mathbf{D}}_x^k\left\{\mathbf{a}_{x1}^k+3\mathbf{a}_{x2}^k+\mathbf{H}_x^k\mathbf{a}_{x3}^k+\mathbf{G}_x^k\mathbf{a}_{x4}^k\right\},$$

where

$$\left(H_x^k\right)_{gg}=\left[\frac{\alpha_{gx}^k\cosh(\alpha_{gx}^k)-m_{gx1}^k(\sinh)}{\sinh(\alpha_{gx}^k)-m_{gx1}^k(\sinh)}\right],\quad \left(H_x^k\right)_{gg'}=0,\ \text{if } g'\neq g.$$

2.8 Analytic Nodal Method (ANM)

In the analytic nodal method (ANM) (Smith, 1979), the transverse-integrated nodal equations (2.15) are solved analytically. An original formulation given in the thesis of K. Smith (1979) is bulky, we will present an alternative algorithm proposed by L. Pogosbekyan and recently incorporated into the SKETCH-N code (Zimin et al., 1999). The algorithm is based on the matrix function theory. A brief introduction to the matrix function theory is given in (Golub and Van Loan, 1996), more details can be found in (Gantmacher, 1959), (Pease and Iii, 1965). Several applications of the matrix functions to neutron diffusion and transport theory are described in textbook (Shikhov and Troyanskii, 1983). The general solution of the Eq. (2.15) is given as:

$$\Phi_x^k(\xi) = \mathbf{sn}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]\mathbf{c1}_x^k + \mathbf{cs}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]\mathbf{c2}_x^k + \mathbf{R}_x^k(\xi), \tag{2.31}$$

where:

$\mathbf{c1}_x^k$ and $\mathbf{c2}_x^k$ are constant vectors determined by boundary conditions;

$\mathbf{R}_x^k(\xi)$ is a particular solution of Eq. (2.15) depending on the trasnverse leakage shape;

$\mathbf{sn}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]$ and $\mathbf{cs}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]$ are matrix functions defined through the buckling matrix $(\mathbf{B}^2)_x^k$.

The matrix functions $\mathbf{sn}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]$ and $\mathbf{cs}\left[\sqrt{(\mathbf{B}^2)_x^k}\xi\right]$ are defined via their scalar analogs:

$$\mathrm{sn}\left[\sqrt{B^2}\xi\right] = \begin{cases} \sinh\left[B\xi\right], & \text{if } B \geq 0 \\ \sin\left[|B|\xi\right], & \text{if } B < 0 \end{cases};$$

$$\mathrm{cs}\left[\sqrt{B^2}\xi\right] = \begin{cases} \cosh\left[B\xi\right], & \text{if } B \geq 0 \\ \cos\left[|B|\xi\right], & \text{if } B < 0 \end{cases}.$$

For the quadratic leakage approximation, the particular solution $\mathbf{R}_x^k(\xi)$ is expressed in terms of the transverse leakage expansion coefficients as:

$$\mathbf{R}_x^k(\xi) = \sum_{i=0}^{2} P_i(\xi)\mathbf{c}_{xi}^k,$$

where the expansion coefficients are:

$$\mathbf{c}_{x0}^k = -\left[(\mathbf{B}^2)_x^k\right]^{-1}\hat{\mathbf{s}}_{x0}^k - 3\left[(\mathbf{B}^2)_x^k\right]^{-2}\hat{\mathbf{s}}_{x2}^k,$$

$$\mathbf{c}_{x1}^k = -\left[(\mathbf{B}^2)_x^k\right]^{-1}\hat{\mathbf{s}}_{x1}^k,$$

$$\mathbf{c}_{x2}^k = -\left[(\mathbf{B}^2)_x^k\right]^{-2}\hat{\mathbf{s}}_{x2}^k.$$

To find the constants of the general solution of the homogeneous equation we

need to define the boundary conditions. In the case of a two-node problem, the boundary conditions are given in terms of the node- averaged face-averaged fluxes at the internal face. Considering the node k they are given as:

$$\frac{1}{2}\int_{-1}^{1}\Phi_x^k(\xi)\,d\xi=\overline{\Phi}^k \text{ and } \Phi_x^k(+1)=\Phi_{x+}^k.$$

Using these boundary conditions, the constants $\mathbf{c1}_x^k$ and $\mathbf{c2}_x^k$ are expressed in terms of the node-averaged flux $\overline{\Phi}^k$ and face-averaged flux Φ_{x+}^k. Substituting the analytic solution (2.31) into Fick's law (2.16) we can compute the face-averaged neutron current at the internal interface of the two-node problem. After some trigonometric manipulations, performed using the symbolic manipulator *Mathematica* (Wolfram, 1996), we obtain the following expression for the face-averaged current:

$$\mathbf{J}_{x+}^k=\mathbf{J}_x^k(+1)= \\ -\hat{\mathbf{D}}_x^k\left\{\mathbf{F}_1\left[\left(\mathbf{B}^2\right)_x^k\right]\overline{\mathbf{\Phi}}^k+\mathbf{F}_2\left[\left(\mathbf{B}^2\right)_x^k\right]\mathbf{\Phi}_{x+}^k+\sum_{i=0}^{2}\mathbf{F}_{i+3}\left[\left(\mathbf{B}^2\right)_x^k\right]\hat{\mathbf{s}}_{xi}^k\right\}, \quad (2.32)$$

where $\mathbf{F}_1\left[\left(\mathbf{B}^2\right)_x^k\right],\ldots,\mathbf{F}_5\left[\left(\mathbf{B}^2\right)_x^k\right]$ are the matrix functions defined by following scalar functions:

$$\{f_1^{\mathrm{ANM}}(x),\ldots,f_5^{\mathrm{ANM}}(x)\}= \\ \left\{x-\frac{1}{\Gamma^2},\frac{1}{\Gamma},1-\frac{1-\Gamma}{\Gamma^2 x},\frac{1-\Gamma}{\Gamma x},\frac{\Gamma(3+x)-3}{\Gamma^2x^2}\right\}, \quad (2.33)$$

where:

$$\Gamma(x)=\frac{\tanh(\sqrt{x})}{\sqrt{x}}=\begin{cases}\tanh(\sqrt{x})/\sqrt{x}, & \text{if } x\geq 0\\ \tan(\sqrt{|x|})/\sqrt{|x|}, & \text{if } x<0\end{cases}.$$

The face-averaged neutron flux Φ_{x+}^k is the only unknown in (2.32). To determine it, we analyze node $k+1$ in the two-node problem and obtain the analytical solution for this node. The solution procedure mirrors that for node k, differing only in boundary conditions. For the node $k+1$, the boundary conditions

are given as:

$$\frac{1}{2}\int_{-1}^{1}\Phi_x^{k+1}(\xi)\,d\xi = \overline{\Phi}^{k+1} \text{ and } \Phi_x^{k+1}(-1) = \Phi_{x-}^{k+1}.$$

The resulting face-averaged current $\mathbf{J}_{x-}^{k+1}$ is expressed as:

$$\mathbf{J}_{x-}^{k+1} = \mathbf{J}_x^{k+1}(-1) = \hat{\mathbf{D}}_x^{k+1}(\mathbf{F}_1\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\overline{\Phi}^{k+1} + \mathbf{F}_2\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\Phi_{x-}^{k+1} + \sum_{i=0}^{2}(-1)^i\mathbf{F}_{i+3}\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\hat{\mathbf{s}}_{xi}^{k+1})$$

Applying continuity conditions at the interface:

$$\mathbf{\Phi}_{x+}^{k} = \mathbf{\Phi}_{x-}^{k+1},$$
$$\mathbf{J}_{x+}^{k} = \mathbf{J}_{x-}^{k+1},$$

we obtain a system of $G \times G$ equations for the face-averaged flux:

$$\left\{\hat{\mathbf{D}}_x^k\mathbf{F}_2\left[\left(\mathbf{B}^2\right)_x^k\right] + \hat{\mathbf{D}}_x^{k+1}\mathbf{F}_2\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\right\}\mathbf{\Phi}_{x+}^{k} = -\hat{\mathbf{D}}_x^k\left\{\mathbf{F}_1\left[\left(\mathbf{B}^2\right)_x^k\right]\overline{\mathbf{\Phi}}^k + \sum_{i=0}^{2}\mathbf{F}_{i+3}\left[\left(\mathbf{B}^2\right)_x^k\right]\hat{\mathbf{s}}_{xi}^k\right\} - \hat{\mathbf{D}}_x^{k+1}\left\{\mathbf{F}_1\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\overline{\mathbf{\Phi}}^{k+1} + \sum_{i=0}^{2}(-1)^i\mathbf{F}_{i+3}\left[\left(\mathbf{B}^2\right)_x^{k+1}\right]\hat{\mathbf{s}}_{xi}^{k+1}\right\}. \quad (2.34)$$

After solving the Eq. (2.34), we can compute the face-averaged current using the expression (2.32).

During our derivation we ignore a problem of computing the matrix functions. There are several ways to make it. A comprehensive review of the methods to compute the matrix exponents gives nineteen dubious ways to make it (Moler and Van Loan, 1978). The authors use the word "dubious" to outline a loss of accuracy in the application of many methods to matrix exponents. In the application to our buckling matrices, which are well-conditioned, none of these methods looks dubious. A need to compute five matrix functions for each reactor node makes a

selection of the most efficient method more important. In the case of two neutron energy groups, our choice of the optimal algorithm is the Lagrange- Sylvester polynomial representation of the matrix functions. In this case, eigenvalues of the buckling matrix are real and eigenvectors form a basis of G-dimensional space. As a result, the matrix function can be computed as (see, for example, Pease and Iii (1965)):

$$\mathbf{F}[\mathbf{B}] = \sum_{g=1}^{G} f(\lambda_g)\mathbf{E}_g[\mathbf{B}, \lambda_g],$$

$$\mathbf{E}_g[\mathbf{B}, \lambda_g] = \prod_{\substack{j=1 \\ j \neq g}}^{G} \frac{\mathbf{B} - \lambda_j \mathbf{I}}{\lambda_g - \lambda_j},$$

where:

$\mathbf{I}$ is the identity matrix

$\mathbf{E}_g[\mathbf{B}, \lambda_g]$ is a projector into the eigensubspace of $\mathbf{B}$ corresponding to eigenvalue λ_g

In the paper (Zimin et al., 1998), we have shown that this way of the matrix function calculations results in the ANM algorithm, which is even faster than the conventional PNM algorithm described in Section 2.6. In the case, when a number of neutron energy groups more than two, eigenvalues of the buckling matrix are complex and computing time of the matrix functions drastically increases. The different algorithm should be considered in this case, for example, Pade approximation of the matrix functions or methods based on Schur decomposition. Because a lack of time we did not implement any these algorithms into the SKETCH code. So, if computing time is important, we recommend using SANM instead of ANM for calculations with more than 2 neutron energy groups.

It is worth to note that the matrix function approach can be applied to other nodal methods. Without going into details we only remark that using the scalar functions given below instead of the ANM functions defined by Eq. (2.33), we

transform ANM into PNM.

$$\{f_1^{PMM}(x),\ldots,f_5^{PMM}(x)\} = \{\frac{1}{5}\left(-66+2x+\frac{2875}{15+x}-\frac{2744}{21+2x}\right), \\ 6-\frac{75}{15+x}, \frac{525+x(15+x)}{(15+x)(10.5+x)}, \frac{5}{15+x}, \frac{21(5+2x)}{5(15+x)(21+2)}\} \quad (2.34)$$

As a result we have an alternative formulation of PNM to that given in Section 2.6. Numerical experiments reported in (Zimin et al., 1998) have shown that the matrix function formulation results in 25% saving in computing time in a comparison with the conventional algorithm described in Section 2.6.

Because $f_i^{ANM}(x) \to f_i^{PNM}(x)$, if $x \to 0, i = 1,\ldots 5$; the PNM functions (2.34) can be also used to remove singularities of the ANM functions (2.33) around zero. This way is used in (Downar et al., 1997b). The approach is somehow awkward, because there is a need to remove singularities only for the eigenvalue, which is close to zero, while in this way it is used for the both eigenvalues. An alternative and more straightforward variant is an application of the Taylor or Pade expansion of the ANM matrix functions as described in (Noh et al., 1997). None of these methods is implemented into the SKETCH-N code. In our limited experience of using the ANM method, we have not found any instabilities of the ANM induced by the singular basic functions.

Chapter 3

Nodal equations for steady-state problems in hexagonal geometry

3.1 Stationary Diffusion Equation.

The core of a nuclear reactor composed of hexagonal prisms is considered. Each prism is treated as an elementary computational volume—a node—with constant properties, which can be defined as:

$$\begin{aligned} V^k(x,y,z): \quad & x \in [-h/2, h/2] \\ & y \in [-y_s(x)/2, y_s(x)/2], \quad y_s(x) = 1/\sqrt{3}\,(\,h - |x|\,) \\ & z \in \left[-\Delta z^k/2, \Delta z^k/2\right] \end{aligned}$$

where h is the node size "across flats."

The multigroup neutron diffusion equation in matrix form is written as:

$$-D^k\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2}\right)\varphi^k(x,y,z)+ \left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\varphi^k(x,y,z)=0, \quad (3.1)$$

where

$\varphi(z,y,z) = col\,\{\varphi_1(z,y,z),\ldots,\varphi_G(z,y,z)\}$ neutron flux density $[cm^{-2}s^{-1}]$;

Σ_r diagonal matrix of removal cross-sections $[cm^{-1}]$;

Σ_s scattering cross-section matrix $[cm^{-1}]$;

k_{eff} effective reactor multiplication factor;

$\chi = col\{\chi_1, \ldots, \chi_G\}$ fission neutron spectrum;

$\nu\Sigma_f = col\{\nu\Sigma_{f1}, \ldots, \nu\Sigma_{fG}\}$ multiplication cross-section matrix $[cm^{-1}]$;

D diagonal matrix of diffusion coefficients $[cm]$;

T denotes the transpose operation.

Integrating equation (3.1) over the node volume yields the balance equation:

$$\sum_{d=x,u,v,z} \overline{J}_d^k S_d + \left[\Sigma_r^k - \Sigma_s^k - \frac{1}{k_{eff}} \chi \left(\nu\Sigma_f^k\right)^T\right] \widehat{\varphi^k} = 0 \tag{3.2}$$

To compute the average neutron flux in the node $\widehat{\varphi^k}$, it is necessary to establish a relationship between the average fluxes and neutron currents on the hexagon faces $\overline{J}$.

The neutron current at the node boundary is represented as the sum of the current computed by the finite-difference formula and a nodal correction:

$$\overline{J}_{gx+}^k = -d_{gx+}^{k,FDM}[\widehat{\varphi}_g^{k+1} - \widehat{\varphi}_g^k] - d_{gx+}^{k,NOD}[\widehat{\varphi}_g^{k+1} + \widehat{\varphi}_g^k], \tag{3.3}$$

where $d_{gx+}^{k,FDM} = \widehat{D}_g^k \widehat{D}_g^{k+1}/(\widehat{D}_g^k + \widehat{D}_g^{k+1})$.

Hence:

$$d_{gx+}^{k,NOD} = -\frac{(d_{gx+}^{k,FDM}[\widehat{\varphi}_g^{k+1} - \widehat{\varphi}_g^k] + \bar{J}_{gx+}^k)}{[\widehat{\varphi}_g^{k+1} + \widehat{\varphi}_g^k]} \tag{3.4}$$

The neutron flux distribution in the reactor is computed using a *nonlinear iterative procedure*. For a given set of nodal coefficients (e.g., zero), the neutron flux is calculated throughout the reactor volume, and the effective multiplication factor is computed. These values are used to compute the neutron currents at the node faces and to update the nodal coefficients via (3.4). With the new nodal coefficients, the neutron flux is recalculated. The procedure repeats until the nodal coefficients converge. Note that using nodal coupling coefficients, compared to the finite-difference method, fully preserves the matrix structure of the problem but makes

the matrix asymmetric even for the one-group case.

The primary task is thus to compute the neutron current at the node boundaries. For this purpose, the three-dimensional diffusion equation is reduced to a one-dimensional form, and an auxiliary one-dimensional problem is solved to compute the neutron current at the boundary between two nodes.

Integrating (3.1) along the z-direction yields a two-dimensional diffusion equation:

$$-D^k\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)\varphi_z^k(x,y)+ \left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\varphi_z^k(x,y)=-L_z^k(x,y), \quad (3.5)$$

where

$\varphi_z^k(x,y)=\dfrac{1}{\Delta z^k}\displaystyle\int_{-\Delta z^k/2}^{\Delta z^k/2}\varphi^k(x,y,z)\mathrm{d}z$ z-averaged neutron flux;

$L_z^k(x,y)=\dfrac{1}{\Delta z^k}\left\{J_z^k(x,y,\Delta z^k/2)-J_z^k(x,y,-\Delta z^k/2)\right\}$ neutron leakage in the z-direction;

$J_z^k(x,y,\pm\Delta z^k/2)$ – z-component of the neutron current at the top/bottom node faces.

The subscript z will be omitted hereafter.

Direct integration along the y-direction does not yield the desired result—a one-dimensional diffusion equation. The hexagonal boundary shape leads to residual terms containing the two-dimensional neutron flux and its y-derivative in the resulting equation. To overcome this difficulty, a conformal mapping of the hexagonal computational domain to a rectangle is used, followed by transverse integration.

Assume a conformal mapping function transforms the hexagon defined on the complex plane $\mathbb{Z}=x+iy$ to a rectangle defined on $\mathbb{W}=u+iv$ (Fig.3.1). A hexagon with size h "across flats" and side length R maps to a rectangle with dimensions $u\in[-a/2,a/2], v\in[0,b]$ (for details, see Appendix A). The real and imaginary parts of $\mathbb{Z}$ (i.e., x and y) form a coordinate grid on the hexagon, while u and v form a grid on the rectangle. The orthogonal grid inside the rectangle generates

a set of orthogonal curves inside the hexagon (Fig.3.1). Notably, the curvilinear coordinate lines intersect the hexagon boundary at right angles, except at the vertices. The properties of conformal mapping ensure that the normal component of the neutron current on the hexagon boundary remains normal for the rectangle.

The Laplace operator is invariant under conformal mapping:

$$\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}=\frac{1}{g^2(u,v)}\left(\frac{\partial^2}{\partial u^2}+\frac{\partial^2}{\partial v^2}\right), \tag{3.6}$$

where $g(u,v)=\mid \frac{\partial \mathbb{Z}}{\partial \mathbb{W}} \mid$ is the distortion factor.

The diffusion equation in the new coordinates becomes:

$$-D^k\left(\frac{\partial^2}{\partial u^2}+\frac{\partial^2}{\partial v^2}\right)\varphi^k(u,v)+ \left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\varphi^k(u,v)g^2(u,v)=-L_z(u,v)g^2(u,v) \tag{3.7}$$

Comparing (3.5) and (3.7), it is clear that they differ only by the distortion factor, which turns a homogeneous cell into a heterogeneous one.

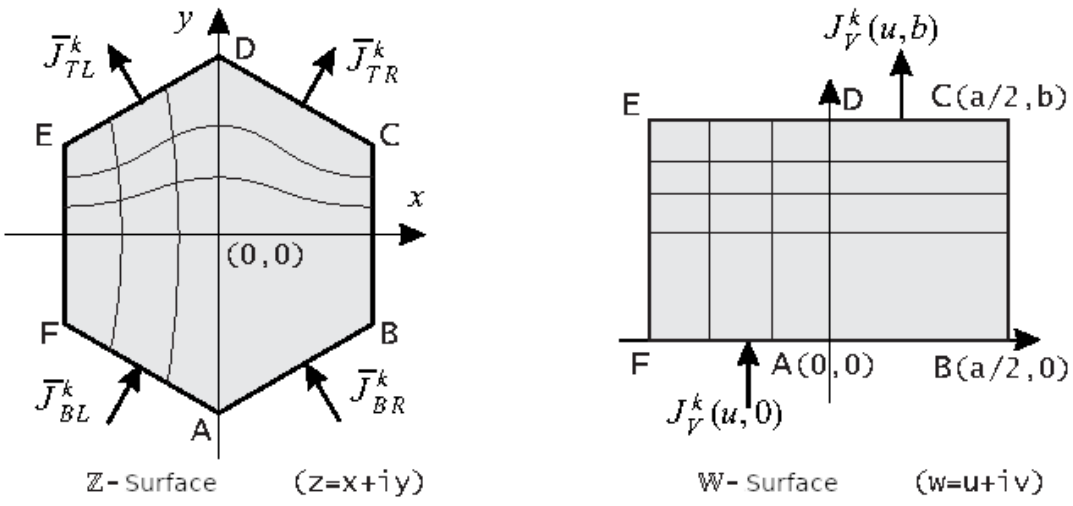


Figure 3.1: Conformal mapping of a hexagon to a rectangle

Define the transversely integrated neutron flux as:

$$\overline{\varphi^k}(u) = \frac{1}{b}\int_0^b \varphi^k(u,v)\mathrm{d}v \tag{3.8}$$

Integrate (3.7) along v:

$$-D^k\frac{\partial^2\overline{\varphi^k}(u)}{\partial u^2} + \left[\Sigma_r^k - \Sigma_s^k - \frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\overline{\varphi^k}(u)\overline{g^2}(u) = \\ -L_z(u)\overline{g^2}(u) - \frac{1}{b}\left[J_v(u,b) - J_v(u,0)\right], \tag{3.9}$$

where

$$\overline{g^2}(u) = \frac{1}{b}\int_0^b g^2(u,v)\varphi(u,v)\mathrm{d}v/\overline{\varphi}(u) \approx \frac{1}{b}\int_0^b g^2(u,v)\mathrm{d}v;$$

$J_v(u,b)$ and $J_v(u,0)$ are the currents in the normal direction of the rectangle at the top and bottom faces;

Using the currents on the hexagon faces and the distortion function, represent $J_v(u,b)$ and $J_v(u,0)$ as:

$$\begin{aligned} J_v(u,b) &= g(u,b)\left[J_{TR}(u)\mu(u) + J_{TL}(u)\mu(-u)\right] \\ J_v(u,0) &= g(u,0)\left[J_{BR}(u)\mu(u) + J_{BL}(u)\mu(-u)\right], \end{aligned} \tag{3.10}$$

where

$\mu(u)$ is the unit step function: $\mu(u) = \begin{cases} 0 & \text{if } u < 0 \\ 1 & \text{if } u > 0 \end{cases}$

$J_{TR}(u), J_{TL}(u), J_{BR}(u), J_{BL}(u)$ are the mapped currents on the hexagon faces (see Fig. 3.1).

Since $g(u,v)$ is symmetric, $g(u,b) = g(u,0)$.

Substituting (3.10) into (3.9) yields the one-dimensional diffusion equation:

$$-D^k\frac{\partial^2\overline{\varphi^k}(u)}{\partial u^2}+\left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\overline{\varphi^k}(u)\overline{g^2}(u)=-L_z(u)\overline{g^2}(u)-$$
$$\frac{1}{b}g(u,0)\left[\left(J_{TR}(u)-J_{BR}(u)\right)\mu(-u)+\left(J_{TL}(u)\mu(u)-J_{BL}(u)\right)\mu(-u)\right], \quad (3.11)$$

To derive the neutron balance equation, integrate (3.11) and multiply by b:

$$b\left[\overline{J}_u(a/2)-\overline{J}_u(-a/2)\right]+b\left[\widehat{\Sigma}_r-\widehat{\Sigma}_s-\frac{1}{k_{eff}}\chi\left(\widehat{\nu\Sigma}_f\right)^T\right]\widehat{\varphi}\int\limits_{-a/2}^{a/2}\overline{g^2}(u)\mathrm{d}u=$$
$$-b\widehat{L}_z\int\limits_{-a/2}^{a/2}\overline{g^2}(u)\mathrm{d}u-\left(\overline{J}_{TR}-\overline{J}_{BR}+\overline{J}_{TL}-\overline{J}_{BL}\right)\int\limits_0^{a/2}g(u,0)\mathrm{d}u \quad (3.12)$$

Integrating the diffusion equation on the $\mathbb{Z}$-plane yields the balance equation:

$$\frac{h}{\sqrt{3}}\left[\overline{J}_x(h/2)-\overline{J}_x(-h/2)\right]+\left[\widehat{\Sigma}_r-\widehat{\Sigma}_s-\frac{1}{k_{eff}}\chi\left(\widehat{\nu\Sigma}_f\right)^T\right]\Phi_{av}\frac{h^2\sqrt{3}}{2}=$$
$$-\widehat{L}_z\frac{h^2\sqrt{3}}{2}-\frac{h}{\sqrt{3}}(\overline{J}_{TR}-\overline{J}_{BR}+\overline{J}_{TL}-\overline{J}_{BL}) \quad (3.13)$$

The following conditions must be satisfied:
condition of equal average fluxes

$$\widehat{\varphi}=\Phi_{av} \quad (3.14)$$

condition of area preservation under conformal mapping

$$ab=\frac{h^2\sqrt{3}}{2} \quad (3.15)$$

Comparing equations (3.12) and (3.13) under constraints (3.14, 3.15), we conclude:

normalization conditions for $g(u,v)$:

$$\int_{-a/2}^{a/2} \overline{g^2}(u)\mathrm{d}u = a \tag{3.16}$$

$$\int_{0}^{a/2} \overline{g}(u,0)\mathrm{d}u = \frac{h}{\sqrt{3}} \tag{3.17}$$

relationship between currents on hexagon and rectangle faces:

$$b\overline{J}_u(a/2) = \frac{h}{\sqrt{3}}\overline{J}_x(h/2) \tag{3.18}$$

Introduce new notation and functions.

Denote the transverse leakage for the rectangle as:

$$L_v(u) = \frac{2}{3h}\left[(J_{TR}(u) - J_{BR}(u))\mu(u) + (J_{TL}(u) - J_{BL}(u))\mu(-u)\right] = L_v^R(u)\mu(u) + L_v^L(u)\mu(-u) \tag{3.19}$$

where

$L_v^R(u) = \frac{2}{3h}\left(J_{TR}(u) - J_{BR}(u)\right)$ – transverse leakage in the right half of the rectangle,

$L_v^L(u) = \frac{2}{3h}\left(J_{TL}(u) - J_{BL}(u)\right)$ – transverse leakage in the left half of the rectangle.

For brevity, introduce an auxiliary function:

$$s(u) = \frac{1}{b}\frac{3h}{2}g(u,0) \tag{3.20}$$

with the normalization $\frac{1}{a}\int_{-a/2}^{a/2} s(u)\mu(u)\mathrm{d}u = 1$.

Introduce a new dimensionless coordinate $\xi = 2u/a$. Then equation (3.11) is

defined on the interval $[-1, 1]$ and, with the new notation, becomes:

$$-\frac{4}{a^2}D^k\frac{\partial^2\overline{\varphi^k}(\xi)}{\partial\xi^2}+\left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\overline{\varphi^k}(\xi)\overline{g^2}(\xi)=-L_z(\xi)\overline{g^2}(\xi)- s(\xi)\left[L_v^R(\xi)\mu(\xi)+L_v^L(\xi)\mu(-\xi)\right] \quad (3.21)$$

Solving equation (3.21) for two adjacent cells establishes a relationship between the current at the cell boundary and the average neutron fluxes. This requires computing the distortion factor $g(u, v)$ of the conformal mapping from a hexagon to a rectangle.

3.2 Polynomial Method for Solving the Stationary Diffusion Equation.

Section 3.1 derived the stationary neutron balance equation (3.21):

$$-\frac{4}{a^2}D^k\frac{\partial^2\overline{\varphi^k}(\xi)}{\partial\xi^2}+\left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]\overline{\varphi^k}(\xi)\overline{g^2}(\xi)= - L_z(\xi)\overline{g^2}(\xi)-s(\xi)\left[L_v^R(\xi)\mu(\xi)+L_v^L(\xi)\mu(-\xi)\right]$$

where $\xi \in [-1, 1]$.

To simplify subsequent formulas, we introduce new notation:

$B^{2^k}=\frac{a^2}{4}D^{k^{-1}}\left[\Sigma_r^k-\Sigma_s^k-\frac{1}{k_{eff}}\chi\left(\nu\Sigma_f^k\right)^T\right]$ – buckling of the k-th cell;

$s_v^k(\xi)=\frac{a^2}{4}D^{k^{-1}}s(\xi)L_v(\xi)$ – transverse leakage in the v-direction;

$s_z^k(\xi)=\frac{a^2}{4}D^{k^{-1}}L_z(\xi)\overline{g^2}(\xi)$ – transverse leakage in the z-direction.

This yields a simplified form of the neutron balance equation:

$$-\frac{\partial^2\varphi^k(\xi)}{\partial\xi^2}+B^{2^k}\overline{g^2}(\xi)\varphi^k(\xi)=-s_v^k(\xi)-s_z^k(\xi) \quad (3.22)$$

To solve this equation, we use a modified version of the polynomial nodal method described in (Zimin and Ninokata, 1998). The original method used

Table 3.1: Coefficient values p_i^j.

$i\backslash j$	0	1	2	3	4
0	1				
1		1			
2	-0.29417		1.29417		
3		-1.14719		2.14719	
4	0.20715		-2.86636		3.65921

Legendre polynomials orthogonal on the interval $[-1,1]$ to solve the diffusion equation in Cartesian coordinates. To improve accuracy by better accounting for the distortion function, we constructed a set of polynomials orthogonal on $[-1,1]$ with weight $\overline{g^2}(\xi)$:

$$\int_{-1}^{1} P_i(\xi)\overline{g^2}(\xi)P_j(\xi)\mathrm{d}\xi = \begin{cases} 0 & \text{if } i \neq j \\ N_i & \text{if } i = j \end{cases} \quad \text{and } P_i(1) = 1 \tag{3.23}$$

The specific form of the polynomials $P_i(\xi) = \sum_{j=0}^{i} p_i^j \xi^j$ was obtained numerically using the Gram-Schmidt orthogonalization procedure and is presented in Table 3.1.

The transversely integrated neutron flux is represented as an expansion in polynomials up to the fourth degree:

$$\overline{\varphi}^k(\xi) = \sum_{i=0}^{4} a_{ui}^k P_i(\xi) = \widehat{\varphi}^k + \sum_{i=1}^{4} a_{ui}^k P_i(\xi), \tag{3.24}$$

where

$a_{u0}^k = \frac{1}{N_0}\int_{-1}^{1} \overline{\varphi}^k(\xi)\overline{g^2}(\xi)\mathrm{d}\xi = \widehat{\varphi}^k$

$a_{ui}^k = col\{a_{ui_1}^k, \ldots, a_{ui_G}^k\}$

The transverse leakage is approximated by a second-degree polynomial:

$$s_{vz}^k(\xi) = \sum_{i=0}^{2} s_{vzi}^k P_i(\xi), \tag{3.25}$$

where $s^k_{vzi} = \dfrac{1}{N_i} \displaystyle\int_{-1}^{1} \left\{ s^k_v(\xi) + s^k_z(\xi) \right\} P_i(\xi)\, \mathrm{d}\xi$.

To compute the expansion coefficients of the transverse leakage s^k_{vzi}, it is necessary to know the form of the transverse leakages in the z- and v-directions. The average leakage values are determined by the average face currents, while additional assumptions must be made about the leakage profile.

3.3 Approximation of Transverse Leakage Shape

In this nodal method, as in most others, average neutron currents across faces are calculated. Consequently, only the mean value of transverse leakage is known. As the development of nodal methods has shown, calculation accuracy can be significantly improved by making additional assumptions about the transverse leakage shape. Let us attempt to construct such assumptions.

Consider a two-dimensional region and assume that the neutron flux density can be represented as:

$$\varphi(x, y) = X(x)Y(y).$$

Under this assumption, the transverse leakage can be written as:

$$L_y(x) = -\frac{1}{\Delta y} D \frac{dY(y)}{dy}\bigg|_{\Delta y} X(x) = const \cdot DX(x)$$

Using neutron current and flux continuity conditions, we obtain the following continuity conditions for transverse leakage and its derivative:

$$\frac{L^k_y(x)}{D^k}\bigg|_{x^k + \frac{\Delta x^k}{2}} = \frac{L^{k+1}_y(x)}{D^{k+1}}\bigg|_{x^{k+1} - \frac{\Delta x^{k+1}}{2}} \tag{3.26}$$

$$\frac{dL^k_y(x)}{dx}\bigg|_{x^k + \frac{\Delta x^k}{2}} = \frac{dL^{k+1}_y(x)}{dx}\bigg|_{x^{k+1} - \frac{\Delta x^{k+1}}{2}} \tag{3.27}$$

The boundary condition given by $D\dfrac{dX(x)}{dx}\bigg|_S = -\gamma X(x)|_S$ leads to a similar

boundary condition for transverse leakage:

$$D\frac{dL_y(x)}{dx}\bigg|_S = -\gamma L_y(x)|_S \tag{3.28}$$

Relations (3.26, 3.27) for internal boundaries and (3.28) for external boundaries are used to compute the coefficients b_i^k of the parabolic approximation for axial transverse leakage:

$$L_z^k(u) = \sum_{i=0}^{2} b_{zi}^k P_i(u). \tag{3.29}$$

For radial transverse leakage $L_v(u)$, the mean leakage values over halves are known, which represents an advancement compared to rectangular geometry and is comparable to quadratic leakage shape approximation. Accordingly, the radial transverse leakage shape for internal cells can be expressed as:

$$L_v(u) = \frac{3h}{2b} g(u,0)(\overline{L}_v^R \mu(u) + \overline{L}_v^L \mu(-u)) \tag{3.30}$$

where

$$\overline{L}_v^R = \frac{2}{3h}\left(\overline{J}_{TR} - \overline{J}_{BR}\right) \quad \overline{L}_v^L(u) = \frac{2}{3h}\left(\overline{J}_{TL} - \overline{J}_{BL}\right)$$

Studies have shown that this approximation yields relatively large errors when calculating reactor models without reflectors. To improve accuracy, we introduce additional assumptions about the transverse leakage shape for boundary cells.

Consider a cell with a right external boundary (Fig. 3.2). Assume that neutron currents along the hexagonal side faces adjacent to the external boundary are approximated by a straight line:

$$J_{TR}^k(x') = \bar{J}_{TR}^k + l_R^k\,(x' - x_0'), \tag{3.31}$$

where the constant l_R^k is determined using boundary conditions (3.28). In this case, the transverse leakage is written as:

$$L_v^R(u) = \frac{3h}{2b} g(u,0)\{\overline{L}_v^R + l_R^k[\int_0^u g(u',0)\mathrm{d}u' - u_0']\} \tag{3.32}$$

or in dimensionless coordinates:

$$L_v^R(\xi) = \frac{3h}{2b} g(\xi, 0)\{\overline{L}_v^R + Rl_R^k[\int_0^\xi g(\tau, 0)\mathrm{d}\tau - \xi_0]\} \tag{3.33}$$

The constant $\xi_0 = u_0'/R$ is chosen based on preserving the mean value of radial transverse leakage in the right half:

$$\xi_0 = \int_0^1 g(\xi, 0) \left(\int_0^\xi g(\tau, 0)\mathrm{d}\tau \right) \mathrm{d}\xi \,/ \int_0^1 g(\xi, 0)\mathrm{d}\xi \tag{3.34}$$

3.4 Two-Cell Problem

To compute the neutron current at the boundary between two nodes, we consider an auxiliary system consisting of two adjacent nodes k and $k+1$. In each node, the neutron flux is described by a diffusion equation of the form (3.22). The problem has $4 \times 2 \times G$ unknowns.

For each cell, we write:

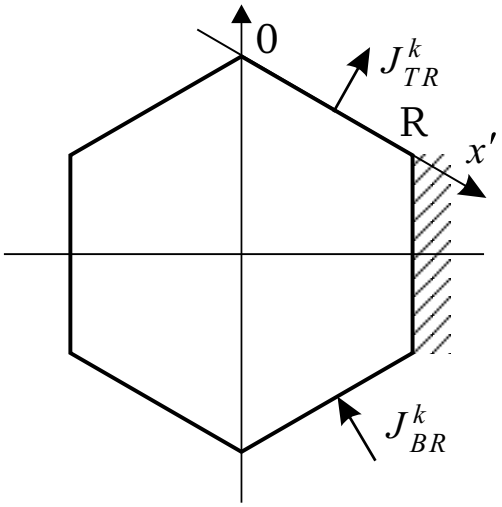


Figure 3.2: Hexagonal cell with right external boundary

- *Neutron balance equations (zeroth moment)* (total $2G$):

$$c_2^J a_{u2}^k + c_4^J a_{u4}^k = B^{2^k}\widehat{\varphi}^k + s_{vz0}^k \tag{3.35}$$

- *Second-order moment-weighting equations* (total $2G$):

$$c_{22} a_{u2}^k + c_{42}^{-1} a_{u4}^k = B^{2^k} a_{u2}^k + s_{vz2}^k \tag{3.36}$$

- *First-order moment-weighting equations* (total $2G$):

$$c_{31}^{-1} a_{u3}^k = B^{2^k} a_{u1}^k + s_{vz1}^k \tag{3.37}$$

- *Neutron flux continuity equation at the common boundary of two cells* (total G):

$$\widehat{\varphi}^k + a_{u1}^k + a_{u2}^k + a_{u3}^k + a_{u4}^k = \widehat{\varphi}_{k+1} - a_{u1}^{k+1} + a_{u2}^{k+1} - a_{u3}^{k+1} + a_{u4}^{k+1} \tag{3.38}$$

- *Neutron current continuity equation at the common boundary of two cells* (total G):

$$\begin{aligned} -\widehat{D}^k[c_1^J a_{u1}^k + c_2^J a_{u2}^k + c_3^J a_{u3}^k + c_4^J a_{u4}^k] = \\ -\widehat{D}_{k+1}[c_1^J a_{u1}^{k+1} - c_2^J a_{u2}^{k+1} + c_3^J a_{u3}^{k+1} - c_4^J a_{u4}^{k+1}] \end{aligned} \tag{3.39}$$

Where the following notations are used:

$$c_i^J = \left.\frac{dP_i(\xi)}{d\xi}\right|_{\xi=1}, \quad i = 1, \ldots, 4.$$

$$c_{22} = \frac{1}{N_2}\int_{-1}^{1} P_2(\xi)\frac{d^2P_2(\xi)}{d\xi^2}\mathrm{d}\xi; \quad c_{42} = \frac{N_2}{\displaystyle\int_{-1}^{1} P_2(\xi)\frac{d^2P_4(\xi)}{d\xi^2}\mathrm{d}\xi};$$

$$c_{31} = \frac{N_1}{\displaystyle\int_{-1}^{1} P_1(\xi)\frac{d^2P_3(\xi)}{d\xi^2}\mathrm{d}\xi}$$

$\widehat{D}^k = \dfrac{2D^k}{\Delta u^k}$ - dimensionless diffusion coefficient.

The values of coefficients c_i^J, c_{22}, c_{42} and c_{31} were obtained numerically; for calculating neutron currents in the axial direction, these coefficients were derived using Legendre polynomials.

From (3.36), we express the fourth coefficient:

$$a_{u4}^k = c_{42}[(B^{2^k} - c_{22})a_{u2}^k + s_{vz2}^k] \tag{3.40}$$

and substitute it into the balance equation (3.35). This yields G equations for the second expansion coefficient of the neutron flux:

$$[c_2^J + c_4^J c_{42}(B^{2^k} - c_{22})]a_{u2}^k = B^{2^k}\widehat{\varphi}^k + s_{vz0}^k - c_4^J c_{42} s_{vz2}^k \tag{3.41}$$

Thus, to compute the even coefficients of the neutron flux expansion, it is sufficient to calculate the inverse matrix of size GxG for the second coefficient and, using the obtained values, compute the fourth coefficients via (3.40).

By substituting the expression for the third approximation coefficient (3.37) into the flux and current continuity equations, we obtain $2G$ equations for calculating the first coefficient:

$$\{I + c_{31}B^{2^{k+1}}\}a_{u1}^{k+1} + \{I + c_{31}B^{2^k}\}a_{u1}^k = \{\widehat{\varphi}^{k+1} + a_{u2}^{k+1} + a_{u4}^{k+1}\} - \{\widehat{\varphi}^k + a_{u2}^k + a_{u4}^k\} - c_{31}\{s_{vz1}^k + s_{vz1}^{k+1}\} \tag{3.42}$$

$$\widehat{D}^{k+1}\{I + c_{31}c_3^J B^{2^{k+1}}\}a_{u1}^{k+1} - \widehat{D}^k\{I + c_{31}c_3^J B^{2^k}\}a_{u1}^k = \widehat{D}^{k+1}\{c_2^J a_{u2}^{k+1} + c_4^J a_{u4}^{k+1} - c_1 s_{vz1}^{k+1}\} + \widehat{D}^k\{c_2^J a_{u2}^k + c_4^J a_{u4}^k - c_1 s_{vz1}^k\} \tag{3.43}$$

After solving the system of $2G$ equations and computing the third coefficients, we obtain the complete set of coefficients for approximating the neutron flux in two adjacent cells.

The neutron current between cells is calculated by:

$$\overline{J}^k_{x+} = \frac{b}{R}\overline{J}^k_{u+} = -\frac{b}{R}\widehat{D}^k\{\sum_{i=1}^{4} c_i^J a^k_{ui}\}, \tag{3.44}$$

where:

$\overline{J}^k_{x+}$ is the average neutron current on the right face of the hexagon;

$\overline{J}^k_{u+}$ is the average neutron current on the right face of the rectangle.

3.5 Boundary with External Medium

Let us consider as an example the right face of the hexagon. Boundary condition (BC) at the right face of the hexagon can be written in the form

$$c_{BC}\overline{J}_x(+h/2) = -f_{BC}\varphi_x(+h/2) \tag{3.45}$$

where

$\overline{J}_x(+h/2)$ is the neutron current at the right face the hexagonal node k;

$\varphi_x(+h/2)$ is face-averaged neutron flux at the right face of the node;

c_{BC},f_{BC} are BC constant, for the vacuum BC $c_{BC} = 1, f_{BC} = 0.5$

The face-averaged neutron currents at the faces of hexagon ($\mathbb{Z}$-plane) $\overline{J}_x(+h/2)$ and rectangles ($\mathbb{W}$-plane) $\overline{J}_u(+a/2)$ have the following relation

$$b\overline{J}_u(+a/2) = \frac{h}{\sqrt{3}}\overline{J}_x(+h/2) \tag{3.46}$$

Below we are working in $\mathbb{W}$-plane, thus we introduce new BC constant $\bar{c}_{BC}$ as follows:

$$\bar{c}_{BC} = c_{BC}\frac{b\sqrt{3}}{h} \approx 1.42510 c_{BC} \tag{3.47}$$

BC equation (3.45) in $\mathbb{W}$-plane can be written as follows

$$\bar{c}_{BC}\overline{J}_u(+a/2) = -f_{BC}\varphi_u(+a/2) \tag{3.48}$$

The neutron flux at the right face of the rectangle $\varphi_u(+a/2)$ can be written as

$$\varphi_u(+a/2) = \widehat{\varphi}^k + a^k_{u1} + a^k_{u2} + a^k_{u3} + a^k_{u4} \tag{3.49}$$

and the neutron current $\overline{J}_u(+a/2)$ is given as

$$\overline{J}_u(+a/2) = -\widehat{D}^k[c^J_1 a^k_{u1} + c^J_2 a^k_{u2} + c^J_3 a^k_{u3} + c^J_4 a^k_{u4}] \tag{3.50}$$

In the expressions above we have two unknowns the 1^{st} and the 3^{rd} expansion coefficient. The 3^{rd} expansion coefficient can be expressed through the 1^{st} one using 1^{st}-order moment weighting equations as

$$c^{-1}_{31} a^k_{u3} = B^{2^k} a^k_{u1} + s^k_{vz1} \tag{3.51}$$

Substituting Eq.(3.49-3.50) into BC equation (3.48) and using the 1^{st} order moment weighting equation (3.51) we obtain G equations with respect to the 1^{st} expansion coefficient for the boundary face as

$$\begin{aligned} &f_{BC}\{I + c_{31}B^{2^k}\}a^k_{u1} + \overline{c}_{BC}\widehat{D}^k\{I + c_{31}c^J_3 B^{2^k}\}a^k_{u1} = \\ &\quad - f_{BC}\{\widehat{\varphi}^k + a^k_{u2} + a^k_{u4} + c_{31}s^k_{vz1}\} - \overline{c}_{BC}\widehat{D}^k\{c^J_2 a^k_{u2} + c^J_4 a^k_{u4} + c_{31}c^J_3 s^k_{vz1}\} \end{aligned} \tag{3.52}$$

Chapter 4

Neutron kinetics model

4.1 Neutron Kinetics Equations for Cartesian Geometry

In Cartesian geometry, few-group neutron kinetics equations in diffusion approximation are written as

$$\frac{1}{v_g}\frac{\partial\varphi_g(r,t)}{\partial t} = \nabla D_g(r,t)\nabla\varphi_g(r,t) - \Sigma_{rg}(r,t)\varphi_g(r,t) + \sum_{\substack{g'=1\\g'\neq g}}^{G}\Sigma_{sgg'}(r,t)\varphi_{g'}(r,t) + (1-\beta)\chi_g^p\sum_{g'=1}^{G}\nu\Sigma_{fg'}(r,t)\varphi_{g'}(r,t) + \chi_g^d\sum_{j=1}^{J}\lambda_j C_j(r,t), \quad (4.1)$$

$$\frac{\partial C_j(r,t)}{\partial t} = \beta_j\sum_{g=1}^{G}\nu\Sigma_{fg}(r,t)\varphi_g(r,t) - \lambda_j C_j(r,t), g = 1,\ldots,G; j = 1,\ldots,J; \quad (4.2)$$

where

to shorter notations we define x, y, z coordinates as r;

j is the delayed neutron precursor group index;

J is the total number of delayed precursor groups;

χ_g^p is the prompt neutron fission spectrum in group g;

χ_g^d is the delayed neutron fission spectrum in group g;

β_j is the delayed neutron yield fraction, group j;

$\beta = \sum_{j=1}^{J} \beta_j$ is the total yield of delayed neutrons per fission;

λ_j is the delayed neutron decay constant, group j;

C_j is the delayed neutron precursor group concentration.

the other notations are given in Section 2.1.

We convert the equations (4.1-4.2) into a matrix form as

$$\mathbf{v}^{-1}\frac{\partial\varphi(r,t)}{\partial t} = \nabla\mathbf{D}(r,t)\nabla\varphi(r,t) - \mathbf{\Sigma}_r(r,t)\varphi(r,t) + \Sigma_s(r,t)\varphi(r,t) + (1-\beta)\chi^p\nu\Sigma_f^T(r,t)\varphi(r,t) + \chi^d\sum_{j=1}^{J}\lambda_j C_j(r,t), \quad (4.3)$$

$$\frac{\partial C_j(r,t)}{\partial t} = \beta_j\nu\Sigma_f^T(r,t)\varphi(r,t) - \lambda_j C_j(r,t), \quad j = 1,...,J;, \quad (4.4)$$

where

$\mathbf{v}^{-1} = \mathrm{diag}\{v_1^{-1},\ldots,v_G^{-1}\}$;

$\chi^p = \mathrm{col}\{\chi_1^p,\ldots,\chi_G^p\}$;

$\chi^d = \mathrm{col}\{\chi_1^d,\ldots,\chi_G^d\}$;

the other notations are given in Section 2.1.

The system of equations (4.3-4.4) is completed by the boundary conditions (2.5) and initial conditions

$$\varphi(r,0) = \varphi_0(r)\,; \quad C_j(r,0) = C_{j0}(r)\,.$$

If we start from the results of the steady-state calculations, $\varphi_0(r)$ is solution of the steady-state eigenvalue problem (2.3-2.4) and the steady-steady state concentration of the delayed neutron precursors is computed as

$$C_{j0}(r) = \frac{\beta_j}{\lambda_j}\, \nu\Sigma_f^T(r,t)\, \varphi_0(r) \tag{4.5}$$

In the transient calculations, we also normalise the multiplication cross section on the value of the steady-state multiplication factor as

$$\nu\Sigma_f(r,t) = \nu\Sigma_f(r,t)/k_{eff}\,.$$

To obtain a numerical solution of the neutron kinetics problem we need to perform time and space discretization. Let us start with discretization in time. First, we analytically integrate the delayed neutron precursors equations (4.4) assuming a linear dependence of the fission source $\nu\Sigma_f^T(r,t)\ \varphi(r,t)$ at the time interval (Stacey, 1969). As a result we have

$$\begin{aligned} C_j(r,t+\Delta t) = {} & C_j(r,t)\exp(-\lambda_j\ \Delta t) + \\ & \frac{\beta_j}{\lambda_j}\left\{\nu\Sigma_f^T(r,t)\ \varphi(r,t)\left[\frac{1-\exp(-\lambda_j\ \Delta t)}{\lambda_j\ \Delta t} - \exp(-\lambda_j\ \Delta t)\right] + \right. \\ & \left. \nu\Sigma_f^T(r,t+\Delta t)\ \varphi(r,t+\Delta t)\left[1 - \frac{1-\exp(-\lambda_j\ \Delta t)}{\lambda_j\ \Delta t}\right]\right\} \end{aligned} \tag{4.6}$$

Using the fully-implicit scheme for neutron flux equation (4.3) and the above expression (4.6) for the delayed neutron precursors we obtain

$$\begin{aligned} & \{-\nabla\mathbf{D}(r,t+\Delta t)\nabla + \mathbf{\Sigma}_r(r,t+\Delta t) + \mathbf{v}^{-1}\frac{1}{\Delta t} - \\ & \qquad \Sigma_s(r,t+\Delta t) - \chi^{NEW}\nu\Sigma_f^T(r,t+\Delta t)\}\cdot\varphi(r,t+\Delta t) = \\ & \left(\mathbf{v}^{-1}\frac{1}{\Delta t} + \chi^{OLD}\nu\Sigma_f^T(r,t)\right)\varphi(r,t) + \chi^d\sum_{j=1}^{J}\lambda_j C_j(r,t)\exp(-\lambda_j\Delta t), \end{aligned} \tag{4.7}$$

where

$$\chi^{NEW} = (1-\beta)\chi^P + \sum_{j=1}^{J} \beta_j \chi^d \left(1 - \frac{1-\exp(-\lambda_j \Delta t)}{\lambda_j \Delta t}\right);$$

$$\chi^{OLD} = \sum_{j=1}^{J} \beta_j \chi^d \left(\frac{1-\exp(-\lambda_j \Delta t)}{\lambda_j \Delta t} - \exp(-\lambda_j \Delta t)\right).$$

Please, note that Eq. (4.6-4.7) are decoupled, at each time step we first solve the neutron flux equation (4.7), which use only delayed neutron precursors from the previous time step and then update the delayed neutron precursors using Eq. (4.6).

The time-discretized equations (4.6-4.7) are discretized in space using the nonlinear iteration procedure described in Section 2.3. The coarse-mesh finite-difference method applied to Eq. (4.6-4.7) yields

$$\sum_{u=x,y,z} F_u^k \left\{ -[\widetilde{\mathbf{D}}_u^k + \widetilde{\mathbf{D}}_u^k(nod)]\overline{\Phi}^{k+1}(t+\Delta t) + \right.$$
$$\left. [\widetilde{\mathbf{D}}_u^k - \widetilde{\mathbf{D}}_u^k(nod) + \widetilde{\mathbf{D}}_u^{k-1} + \widetilde{\mathbf{D}}_u^{k-1}(nod)]\overline{\Phi}^k(t+\Delta t) - [\widetilde{\mathbf{D}}_u^{k-1} - \widetilde{\mathbf{D}}_u^{k-1}(nod)]\overline{\Phi}^{k-1}(t+\Delta t) \right\} +$$
$$\left(\Sigma_r^k + \mathbf{v}^{-1}\frac{1}{\Delta t} - \Sigma_s^k \overline{\Phi}^k V^k - \chi^{NEW}(\nu\Sigma_f^k)^T\right)\overline{\Phi}^k(t+\Delta t)\, V^k =$$
$$\left(\mathbf{v}^{-1}\frac{1}{\Delta t} + \chi^{OLD}(\nu\Sigma_f^k)^T(t)\right)\overline{\Phi}^k(t)\, V^k + \chi^d V^k \sum_{j=1}^{J} \lambda_j \overline{C}_j^k(t)\exp(-\lambda_j \Delta t) \quad (4.8)$$

$$\overline{C}_j^k(t+\Delta t) = \overline{C}_j^k(t)\exp(-\lambda_j \Delta t) +$$
$$\frac{\beta_j}{\lambda_j}\left\{ \left(\nu\Sigma_f^k\right)^T(t)\overline{\Phi}^k(t)\left[\frac{1-\exp(-\lambda_j\,\Delta t)}{\lambda_j\,\Delta t} - \exp(-\lambda_j\,\Delta t)\right] + \right.$$
$$\left. \left(\nu\Sigma_f^k\right)^T(t+\Delta t)\overline{\Phi}^k(t+\Delta t)\left[1 - \frac{1-\exp(-\lambda_j\,\Delta t)}{\lambda_j\,\Delta t}\right]\right\} \quad (4.9)$$

where

all cross sections at the left-hand side are evaluated at the time $(t+\Delta t)$,

$\overline{C}_j^k \equiv \frac{1}{V^k} \int\limits_{-\Delta z/2}^{\Delta z/2} dz \int\limits_{-\Delta y/2}^{\Delta y/2} dy \int\limits_{-\Delta x/2}^{\Delta x/2} dx\, \overline{C}_j^k(x,y,z)$ is node-averaged concentration of the delayed neutron precursors;

the other notations are given in Sections 2.2 and 2.3.

To obtain the nodal equations, we apply at first the transverse-integration procedure to the neutron flux equations (4.7). The result follows

$$\left\{-\mathbf{D}^k \frac{d^2}{dx^2} + \Sigma_r^k + \mathbf{v}^{-1}\frac{1}{\Delta t} - \Sigma_s^k - \chi^{NEW}(\nu\Sigma_f^k)^T\right\}\Phi_x^k(x,t+\Delta t) = \\ -\mathbf{S}_x^k(x,t+\Delta t) + \mathbf{Q}_x^k(x,t), \quad (4.10)$$

where

we again dropped time index $(t+\Delta t)$ of the cross sections at the left hand side and

$$\mathbf{Q}_x^k(x,t) = \left(\mathbf{v}^{-1}\tfrac{1}{\Delta t} + \chi^{OLD}(\nu\Sigma_f^k)^T(t)\right)\Phi_x^k(x,t) + \chi^d \textstyle\sum_{j=1}^J \lambda_j C_{jx}^k(x,t)\exp(-\lambda_j\Delta t);$$

$C_{jx}^k(x) \equiv \frac{1}{F_x^k}\int_{-\Delta y/2}^{\Delta y/2} dy \int_{-\Delta z/2}^{\Delta z/2} dz C_j^k(x,y,z)$ is the transverse-integrated concentration of the delayed neutron precursors, group j.

Straightforward application of the nodal methods to Eq. (4.10) requires knowledge of the higher-order expansion coefficients of the flux and precursor concentration from the previous time step. To reduce the computer memory requirements we follow a trick proposed by Engrand et al. (1994). The neutron flux equation (4.10) is led to the form of the steady-state equation (2.13) grouping all additional terms into the transverse-leakage term $\tilde{\mathbf{S}}_x^k(x,t+\Delta t)$. As a results we have

$$\left\{-\mathbf{D}^k \frac{d^2}{dx^2} + \Sigma_r^k\right\}\Phi_x^k(x,t+\Delta t) = \\ \left\{\Sigma_s^k + \chi^p(\nu\Sigma_f^k)^T\right\}\Phi_x^k(x,t+\Delta t) - \tilde{\mathbf{S}}_x^k(x,t+\Delta t), \quad (4.11)$$

where

$$\tilde{\mathbf{S}}_x^k(x,t+\Delta t) = \mathbf{S}_x^k(x,t+\Delta t) + \mathbf{v}^{-1}\frac{1}{\Delta t}\left\{\Phi_x^k(x,t+\Delta t) - \Phi_x^k(x,t)\right\} - \chi^{OLD}(\nu\Sigma_f^k)^T(t)\Phi_x^k(x,t) - \left\{\chi^{NEW} - \chi^p\right\}(\nu\Sigma_f^k)^T(t+\Delta t)\Phi_x^k(x,t+\Delta t) - \chi^d\sum_{j=1}^{J}\lambda_j C_{jx}^k(x,t)\exp(-\lambda_j\Delta t). \quad (4.12)$$

Equation (4.11) is practically identical to the steady-state equation (2.13). As a result all the nodal methods described in the Chapter 2 are applied in the same way to the neutron kinetics problems. The only difference is the calculation of the transverse-leakage term $\tilde{\mathbf{S}}_x^k(x,t+\Delta t)$.

4.2 Neutron Kinetics Equations for Hexagonal Geometry

The multigroup neutron diffusion kinetics equation in matrix form can be written as:

$$v^{-1}\frac{\partial\varphi(\vec{r},t)}{\partial t} = \nabla D\nabla\varphi(\vec{r},t) - \Sigma_r\varphi(\vec{r},t) + \Sigma_s\varphi(\vec{r},t) + (1-\beta)\chi^p\left(\nu\Sigma_f\right)^T\varphi(\vec{r},t) + \chi^d\sum_{j=1}^{J}\lambda_j C_j(\vec{r},t), \quad (4.13)$$

$$\frac{\partial C_j(\vec{r},t)}{\partial t} = \beta_j\left(\nu\Sigma_f\right)^T\varphi(\vec{r},t) - \lambda_j C_j(\vec{r},t) \quad (4.14)$$

where

j - delayed neutron group index ($j = 1,\ldots,J$);

v^{-1} - diagonal matrix of inverse velocities, $v^{-1} = diag\{v_1^{-1},\ldots,v_G^{-1}\}$;

χ^p - prompt fission neutron spectrum, $\chi^p = col\{\chi_1^p,\ldots,\chi_G^p\}$;

χ^d - delayed neutron spectrum, $\chi^d = col\{\chi_1^d,\ldots,\chi_G^d\}$;

β_j - delayed neutron fraction for group j;

$\beta = \sum_{j=1}^{J} \beta_j$ - total delayed neutron fraction;

λ_j - decay constant for delayed neutron group j;

$C_j(\vec{r}, t)$ - delayed neutron precursor concentration.

Equations (4.13, 4.14) are supplemented with the initial condition

$$\varphi(\vec{r}, 0) = \varphi_0(\vec{r}),$$

which is the solution of the conditional critical steady-state problem. The initial delayed neutron precursor concentrations are calculated as[1]

$$C_{j0}(\vec{r}) = \frac{\beta_j}{\lambda_j} (\nu\Sigma_f)^T \varphi_0(\vec{r}). \tag{4.15}$$

Assuming the neutron flux density varies linearly over the time interval $t \in [t, t + \Delta t]$, we analytically integrate equation (4.14) to obtain:

$$\begin{aligned} C_j(\vec{r}, t + \Delta t) = C_j(\vec{r}, t) \exp(-\lambda_j \Delta t) + \\ \frac{\beta_j}{\lambda_j} \left\{ (\nu\Sigma_f)^T \varphi(\vec{r}, t) \left[-\exp(-\lambda_j \Delta t) + \frac{1 - \exp(-\lambda_j \Delta t)}{\lambda_j \Delta t} \right] + \right. \\ \left. + (\nu\Sigma_f)^T \varphi(\vec{r}, t + \Delta t) \left[1 - \frac{1 - \exp(-\lambda_j \Delta t)}{\lambda_j \Delta t} \right] \right\} \end{aligned} \tag{4.16}$$

Using an implicit time integration scheme for equation (4.13) and expression (4.16), we obtain:

$$\begin{aligned} \left(-\nabla D \nabla + v^{-1} \frac{1}{\Delta t} + \Sigma_r - \Sigma_s - \chi^{new} (\nu\Sigma_f)^T \right) \varphi(\vec{r}, t + \Delta t) = \\ = \left(v^{-1} \frac{1}{\Delta t} + \chi^{old} (\nu\Sigma_f)^T \right) \varphi(\vec{r}, t) + \chi^d \sum_{j=1}^{J} \lambda_j C_j(\vec{r}, t) \exp(-\lambda_j \Delta t), \end{aligned} \tag{4.17}$$

[1] Here and subsequently, we assume that fission cross-sections are normalized to the effective multiplication factor obtained from the steady-state solution: $\Sigma_f(\vec{r}, t) = \Sigma_f(\vec{r}, t)/K_{\text{eff}}$

where we introduce:

$$\chi^{new} = (1-\beta)\chi^p + \sum_{j=1}^{J} \beta_j \chi^d \left(1 - \frac{1-\exp(-\lambda_j \Delta t)}{\lambda_j \Delta t}\right),$$

$$\chi^{old} = \sum_{j=1}^{J} \beta_j \chi^d \left(\frac{1-\exp(-\lambda_j \Delta t)}{\lambda_j \Delta t} - \exp(-\lambda_j \Delta t)\right).$$

Equation (4.17) can be represented in matrix form:

$$\mathtt{A} \cdot x = b \tag{4.18}$$

By solving equation (4.18) at each time step, we obtain the solution to the non-stationary neutron physics problem. The matrix $\mathtt{A}$ is of general form - even in the one-group case it is not symmetric due to nodal coupling coefficients. The solution method will be described in the next section. Here we focus on calculating nodal coefficients in the non-stationary case.

Applying transverse integration to equation (4.17) yields:

$$\begin{aligned}\left\{-D^k \frac{d^2}{du^2} + v^{-1}\frac{1}{\Delta t} + \Sigma_r^k - \Sigma_s^k - \chi^{new}(\nu\Sigma_f^k)^T\right\}\varphi_k(u,t+\Delta t) = \\ = -L_z^k(u,t+\Delta t)\overline{g^2}(u) - \frac{1}{b}\frac{3h}{2}g(u,0)L_v^k(u,t+\Delta t) + \\ + \left(v^{-1}\frac{1}{\Delta t} + \chi^{old}(\nu\Sigma_f^k)^T\right)\varphi_k(u,t) + \chi^d \sum_{j=1}^{J}\lambda_j C_{ju}^k(u,t)\exp(-\lambda_j \Delta t)\end{aligned} \tag{4.19}$$

where $C_{ju}^k(u,t) = \frac{1}{b}\int_0^b \mathrm{d}v \frac{1}{\Delta z}\int_{-\Delta z/2}^{\Delta z/2} C(u,v,z)\mathrm{d}z$ is the transversely-integrated precursor concentration.

Using equation (4.19) for calculating nodal coupling coefficients is problematic since it requires storing the expansion coefficients of both neutron flux and delayed neutron precursors from the previous time step. To overcome this issue, we can apply the method described in (Engrand et al., 1994). By moving terms containing kinetic parameters to the right-hand side, equation (4.19) can be transformed to

a form similar to the stationary case with a modified right-hand side:

$$\left\{-D^k\frac{d^2}{du^2}+\Sigma_r^k-\Sigma_s^k-\chi^p(\nu\Sigma_f^k)^T\right\}\varphi_k(u,t+\Delta t)=$$
$$=-L_z^k(u,t+\Delta t)\overline{g^2}(u)-\frac{1}{b}\frac{3h}{2}g(u,0)L_v^k(u,t+\Delta t)-$$
$$-v^{-1}\frac{1}{\Delta t}\left\{\varphi_k(u,t+\Delta t)-\varphi_k(u,t)\right\}+\chi^{old}(\nu\Sigma_f^k)^T\varphi_k(u,t)+$$
$$+\left\{\chi^{new}-\chi^p\right\}(\nu\Sigma_f^k)^T\varphi_k(u,t+\Delta t)+\chi^d\sum_{j=1}^{J}\lambda_jC_{ju}^k(u,t)\exp(-\lambda_j\Delta t) \quad (4.20)$$

When computing the right-hand side of equation (4.20), we use average values of neutron flux and delayed neutron precursor concentrations. In cases of strong neutron field shape deformations within a single time step, iterative refinement of nodal coupling coefficients may be necessary. This problem can be addressed by reducing the time step size. In the transient tests described in subsequent sections, we performed one nodal coefficient calculation per time step.

4.3 Automatic Time Step Size Control

Most of the modern neutron kinetics codes use an automatic time step control procedure in order to obtain a solution with a given tolerance (Aviles, 1993; Crouzet and Turinsky, 1996; Joo et al., 1996). Moreover, a variable time step size usually decreases a total amount of time steps and respectively computing time. We use the standard time-step doubling strategy (Hairer et al., 1987) in order to estimate a temporal truncation error and to predict a time step size. The procedure is mathematically well-founded and described in details elsewhere (Hairer et al., 1987; Crouzet and Turinsky, 1996), we only shortly outline the main features. Time integration from the time moment t_j is performed on the two temporal meshes: on the fine mesh with the time step size Δt_j and on the coarse mesh with the time step size $2\Delta t_j$ as shown in Fig. 4.1.

As a result at the time moment $t_j+2\Delta t_j$ we have the two solutions: $\Phi_{\Delta t_j}$ and $\Phi_{2\Delta t_j}$, respectively. As a solution we take the node-averaged power distribution because generally we want to control accuracy of the computed power. The local

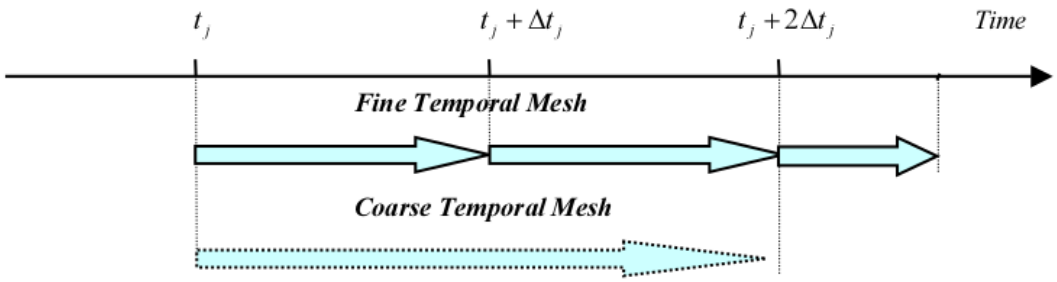


Figure 4.1: Time stepping in the time step doubling technique

error of the fully-implicit scheme can be expressed as follows

$$Error = Const\Delta t^2 + O(\Delta t^3), \tag{4.21}$$

where constant $Const$ is proportional to the 2$^{\text{nd}}$ derivative of the neutron flux.

Applying the Richardson extrapolation for the two obtained solutions and neglecting the higher order terms we can obtain the temporal truncation error of the solution $\Phi_{\Delta t_j}$ as

$$Error = \left\| \Phi_{\Delta t_j} - \Phi_{2\Delta t_j} \right\|. \tag{4.22}$$

A most appropriate norm in the expression 4.22 is L_∞-norm if we wants to held the local errors of the power distribution within a given tolerance. However, an application of L_2-norm demonstrates more robust qualities for the time step size selection (Crouzet and Turinsky, 1996). In addition, the error, which has to be controlled, is a relative one and the expression 4.22 is replaced by the following one

$$Error = \frac{\left\| \Phi_{\Delta t_j} - \Phi_{2\Delta t_j} \right\|_2}{\left\| \Phi_{\Delta t_j} \right\|_2}.$$

Using the expression 4.21 and requiring that the temporal truncation error is equal to the prescribed tolerance we get a prediction of the new time step size as

$$\Delta t^{NEW} = \sqrt{\frac{Tolerance}{Error}} \Delta t_j.$$

A time step acceptance criterion can be written in the form

$$Error \leq Tolerance.$$

If the estimated error satisfies this criterion, the j-th time step is accepted and $\Delta t_{j+1} = \Delta t^{NEW}$ is used as a next time step size. Otherwise, the current time step is rejected and repeated with $\Delta t_j = \Delta t^{NEW}$. In order to avoid repeated time step size rejections, safety parameters are introduced. The new time step size is calculated as

$$\Delta t^{NEW} = \Delta t_j \min \left[facmax, \max \left[facmin, safety \frac{\Delta t^{NEW}}{\Delta t_j} \right] \right],$$

where $facmax$ and $facmin$ are equal to 2 and 0.5; the typical value of the parameter $safety$ is in a range 0.8-0.9.

Performing calculations we often want to obtain a solution at the prescribed time moments. In order to handle this problem without abrupt change in the time step size, an additional modification of the new time step size Δt^{NEW} is done as follows

$$\Delta t^{NEW} = \frac{Time - t_j}{\left| \frac{Time - t_j}{\Delta t^{NEW}} - \varepsilon \right|},$$

where

$Time$ is the prescribed time moment,

ε is a small multiple of the machine roundoff error;

$|\circ|$ denotes the integer ceiling function.

4.4 Point Kinetics Model

Three-dimensional neutron kinetics calculations are time-consuming and a code user usually wants to know an influence of space-dependent effects on transient results. To provide such capabilities, the SKETCH-N code has a point kinetics model, where neutron flux shape is assumed to be a constant during a transient.

Thus, a comparison of the results computed by the three-dimensional model and the point kinetics model can be done using the same code.

The basic assumption of the point kinetics model is that the neutron flux vector can be factored into the product of the neutron flux shape, which does not vary in time, and amplitude function as

$$\varphi(r,t) = T(t)\psi(r), \tag{4.23}$$

where the shape function is normalised such that $\langle \mathbf{w}(r), \mathbf{v}^{-1}\psi(r)\rangle = 1$; $\mathbf{w}(r)$ is the weight function and the scalar product operation $\langle \circ, \circ \rangle$ is the integration over the reactor domain and neutron energy groups.

Weight function $\mathbf{w}(r)$ is usually chosen as a solution of the steady-state adjoint eigenvalue problem $\varphi_0^+(r)$. This choice eliminates the first-order errors in the reactivity due to the change of the neutron flux shape (Henry, 1975). A solution of the adjoint eigenvalue problem is performed in the SKETCH-N code before the transient calculations. Substituting the factorization (4.23) into Eq. (4.3-4.4), multiplying by the weight function $\varphi_0^+(r)$ and integrating over the reactor volume yields the point kinetics equations

$$\frac{dT(t)}{dt} = \frac{[\rho - \beta]}{\Lambda} T(t) + \sum_{j=1}^{J} \lambda_j c_j(t), \tag{4.24}$$

$$\frac{dc_j(t)}{dt} = \frac{\beta_j}{\Lambda} T(t) - \lambda_j c_j(t),\ j = 1, ..., J; \tag{4.25}$$

where the point kinetics parameters are defined as

$$\rho = \rho(t) = \left\langle \varphi_0^+(r), [\nabla \mathbf{D}(r,t)\nabla - \mathbf{\Sigma}_r(r,t) + \Sigma_s(r,t) + \right.$$
$$\left. \left((1-\beta)\chi^p + \beta\chi^d\right)\nu\Sigma_f^T(r,t)\right]\psi(r)\Big\rangle / F;$$

$$\beta_j = \beta_j(t) = \left\langle \varphi_0^+(r), \beta_j \chi^d \nu \Sigma_f^T(r,t)\psi(r)\right\rangle / F;$$

$$\beta = \textstyle\sum_{j=1}^{J} \beta_j;$$

$$\Lambda = \Lambda(t) = \left\langle \varphi_0^+(r), \mathbf{v}^{-1}\psi(r)\right\rangle / F = 1/F;$$

$$F = F(t) = \left\langle \varphi_0^+(r), \left((1-\beta)\chi^p + \beta\chi^d\right)\nu\Sigma_f^T(r,t)\psi(r)\right\rangle;$$

$c_j(t) = \left\langle \varphi_0^+(r), \chi^d C_j(r,t) \right\rangle / F.$

The system of equations (4.24-4.25) is completed by the initial conditions as

$$T(0) = \left\langle \varphi_0^+(r), \mathbf{v}^{-1}\varphi_0(r) \right\rangle;$$

$$c_j(t) = \left\langle \varphi_0^+(r), \chi^d C_j(r,0) \right\rangle / F;$$

where the initial steady-state concentration of the delayed neutron precursors is determined by Eq. (4.5). Time discretization of the point kinetics equations is performed by the same fully-implicit scheme with analytical integration of the delayed neutron precursors as in the case of the three-dimensional model (Stacey, 1969). The time-discretized point kinetics equations are

$$T(t+\Delta t) = \frac{T(t)\left\{\frac{1}{\Delta t} + \frac{1}{\Lambda}\sum_{j=1}^{J}\beta_j\left[\frac{1-\exp(-\lambda_j\Delta t)}{\lambda_j\Delta t} - \exp(-\lambda_j\Delta t)\right]\right\} + \sum_{j=1}^{J}\lambda_j c_j(t)\exp(-\lambda_j\Delta t)}{\frac{1}{\Delta t} - \frac{\rho(t)}{\Lambda} + \frac{1}{\Lambda}\sum_{j=1}^{J}\beta_j\left[\frac{1-\exp(-\lambda_j\Delta t)}{\lambda_j\Delta t}\right]}$$

$$c_j(t+\Delta t) = c_j(t)\exp(-\lambda_j\Delta t) + \frac{\beta_j}{\lambda_j\Lambda}\left\{T(t)\left[\frac{1-\exp(-\lambda_j\Delta t)}{\lambda_j\Delta t} - \exp(-\lambda_j\Delta t)\right] + T(t+\Delta t)\left[1 - \frac{1-\exp(-\lambda_j\Delta t)}{\lambda_j\Delta t}\right]\right\}$$

Automatic time step control procedure described in the previous Section 4.3 is also applied for the point kinetics equations.

Before leaving this section we would like to note that the point kinetics parameters in the SKETCH code are generated automatically at each time step from the current values of the macro cross sections. Macro cross sections are recomputed at each time step depending on the control rod position and values of thermal-hydraulics feedbacks. Thus, a code user is liberated from the difficult and time-consuming task of preparation of the point kinetics parameters, which he has to do if he is using the transient analysis code with a separate point kinetics model.

Chapter 5

Iterative solution of the steady-state eigenvalue problems

The nonlinear iteration procedure introduced in Chapter 2 results that a solution of the neutron diffusion equations (2.2-2.4) is decoupled into a global solution of the coarse-mesh finite-difference equations (2.11) and local solutions of the two-node nodal equations. The solution of the nodal equations of the polynomial, semi-analytic and analytic methods is discussed in the Chapter 2, while the iterative solution techniques applied to the equation (2.11) are considered in this Chapter.

5.1 Steady-State Eigenvalue Problem

The coarse-mesh finite-difference equation (2.11) can be written in the matrix form as

$$\mathbf{M}[\Phi, \lambda, \mathbf{T}]\Phi = \frac{1}{\lambda}\mathbf{F}[\mathbf{T}]\Phi, \tag{5.1}$$

where

λ is the eigenvalue, the reactor multiplication factor k_{eff};

$\Phi = \text{col}\{\overline{\Phi}^1, \ldots, \overline{\Phi}^K\}$ is the vector of the node-averaged neutron flux of size KG, K is the total number of the nodes, G is the number of neutron energy groups;

$\mathbf{M}[\Phi, \lambda, \mathbf{T}]$ is the $KG \times KG$ matrix, containing the finite-difference entries of the diffusion, removal and scattering operators; the dependency of the matrix $\mathbf{M}$ on the neutron flux Φ, eigenvalue λ and thermal-hydraulics parameters $\mathbf{T}$ is explicitly stated;

$\mathbf{F}[\mathbf{T}]$ is the $KG \times KG$ matrix, containing the finite-difference entries of the multiplication operator $\chi(\nu\Sigma_f^k)^T$, the matrix depends on the thermal-hydraulics feedbacks $\mathbf{T}$.

A solution of the steady-state eigenvalue problem (5.1) involves several iteration levels as shown in Fig.5.1. A top level is the thermal-hydraulics iterations, where we recompute the macro cross sections and the finite-difference coefficients of the matrices $\mathbf{M}$ and $\mathbf{F}$ depending on the values of the thermal hydraulics feedbacks (fuel temperature, coolant density etc.). The thermal-hydraulics feedbacks depend on the power distribution and respectively on the neutron flux shape. The calculation of the thermal- hydraulics parameters using an internal thermal-hydraulics module is discussed in Chapter 8; an external thermal-hydraulics model can be coupled with the SKETCH-N code using an interface module described in Chapter 9. A next iteration level is the nonlinear iterations, where the nodal coupling coefficients of the matrix $\mathbf{M}$ are updated depending on the node-averaged neutron fluxes and the eigenvalue. The solution of the two-node nodal equations and recalculation of the nodal coupling coefficients are given in Chapter 2.2. With fixed thermal-hydraulics feedbacks and the nodal coupling coefficients the problem (5.1) is reduced to a linear eigenvalue problem as

$$\mathbf{M}\Phi = \frac{1}{\lambda}\mathbf{F}\Phi, \tag{5.2}$$

Sections 5.2 and 5.3 present the outer-inner iteration techniques used for solving Eq. (5.2).

In the code, we combine the thermal-hydraulics and nonlinear iterations into the one level called nonlinear iterations. The nonlinear iterations terminate when the outer iterations solving the linearized eigenvalue problem (5.2) converge in one iteration. A fixed number of outer iterations per nonlinear iteration is used if the convergence of the outer iterations is not reached. The number depends on the problem at hand, an optimal convergence is observed with 8-15 outer iterations per

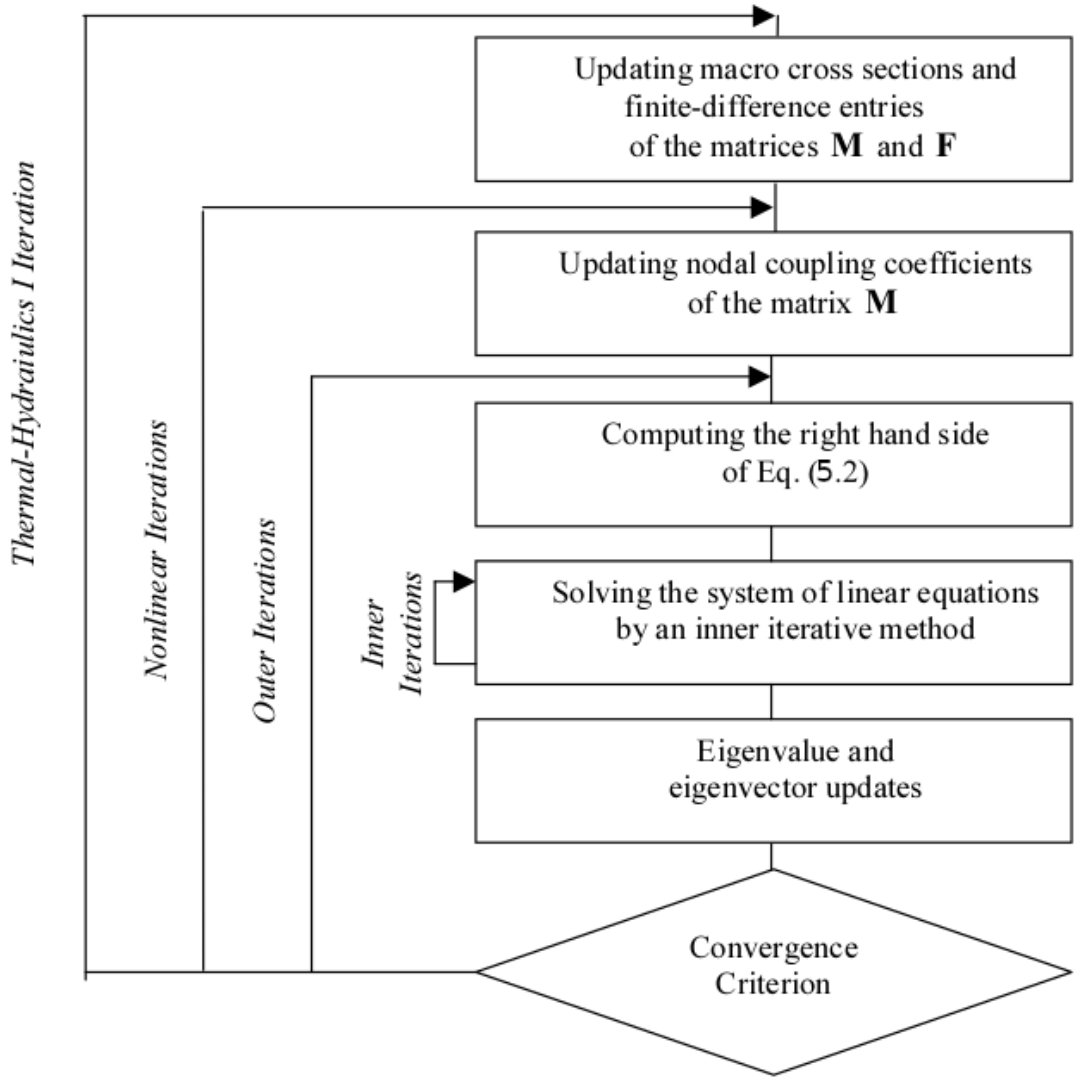


Figure 5.1: Iterative levels of the steady-state eigenvalue problem

nonlinear iteration for performed LWR calculations. An algorithm of the global iterations is given in Fig. 5.2.

Let us make some theoretical remarks on the solution of the linear eigenvalue problem (5.2). In the limit to zero of the spatial mesh size, the nodal coupling coefficients of the matrix $\mathbf{M}$ also approach zero and the matrix $\mathbf{M}$ has the following properties:

- renumbering the vector Φ first by mesh points, second by neutron groups results in the matrix $\mathbf{A}$ with G blocks $\mathbf{M}_g$ size of $K \times K$, which are symmetric;
- diagonal elements of $\mathbf{M}$ are positive;
- off-diagonal elements of $\mathbf{M}$ are nonpositive (negative or zero);
- $\mathbf{M}$ is diagonally dominant;
- $\mathbf{M}$ is irreducible.

Under these conditions, theoretical analysis has been performed using Perron-Frobenious theory of nonnegative matrices, see Section 3.3 in (Wachspress, 1966). It is proved that the problem has simple real eigenvalue λ_1, such that $\lambda_1 > |\lambda_i|$, $i = 2, \ldots$ and the corresponding eigenvector $\Phi_1 > 0$.

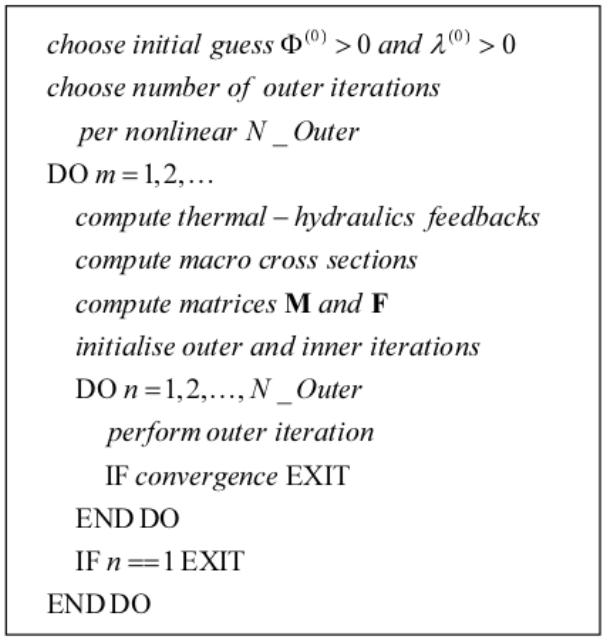


Figure 5.2: Algorithm of the nonlinear iterations of the steady-state eigenvalue problem

5.2 Outer Iterations

A power method (Nakamura, 1977) can be used for solving the generalized eigenvalue problem (5.2). An algorithm of the method is given in Fig. 5.3, where $\mathbf{e} = \mathrm{col}\{1, 1, \ldots, 1\}$ is the unity vector; $\langle \circ, \circ \rangle$ denotes the dot product.

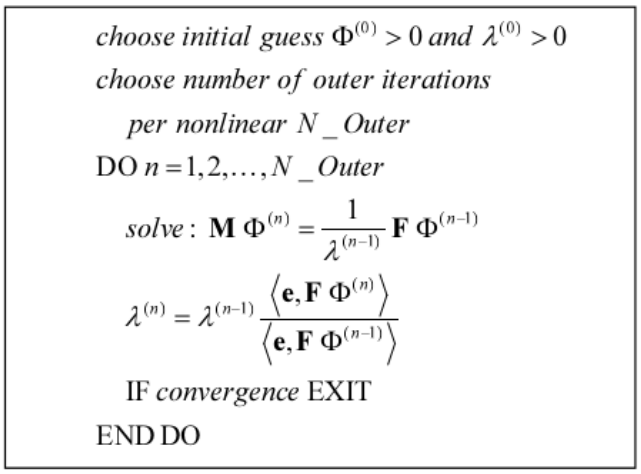


Figure 5.3: Algorithm of the power method applied to the generalized eigenvalue problem

The iterations of the power method are called outer iterations, because a system of linear equations

$$\mathbf{M}\Phi^{(n)} = \frac{1}{\lambda^{(n-1)}}\mathbf{F}\Phi^{(n-1)} \tag{5.3}$$

is usually also solved by an iterative method forming inner iterations. A convergence rate of the power method is determined by the dominance ratio $d = \frac{\lambda_2}{\lambda_1}$, where λ_2 is the second largest eigenvalue of the generalized eigenvalue problem (5.2). Decreasing the dominance ratio d increases the convergence. In practical LWR models, the dominance ratio is close to unity and the convergence may be very slow.

The simplest and the most popular method to improve the convergence of the power method is to move a part of the right hand side of Eq. (5.2) into the left hand side as

$$\left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]\Phi = \frac{1}{\Lambda}\Phi \tag{5.4}$$

$$\frac{1}{\Lambda} = \frac{1}{\lambda} - \frac{1}{\lambda'},$$

where λ' is the chosen eigenvalue shift.

The dominance ratio of the shifted eigenvalue problem (5.4) is given as

$$d' = \frac{\frac{1}{\lambda_1} - \frac{1}{\lambda'}}{\frac{1}{\lambda_2} - \frac{1}{\lambda'}}$$

If λ' is larger than λ_1, the dominance ratio is less than unity and less than the dominance ratio d of the original eigenvalue problem (5.2). If λ' is equal to the eigenvalue λ_1, the dominance ratio is zero, but we may encounter a problem of solving the linear system by inner iterations, because the matrix $\left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]$ is singular. In practice, the eigenvalue shift λ' is computed during the iterations as follows

$$\lambda' = \lambda^{(n)} + \delta\lambda,$$

where $\lambda^{(n)}$ is the current estimate of the eigenvalue and $\delta\lambda$ is the chosen shift parameter.

An optimal choice of the shift parameter is a problem dependent. Decreasing the shift parameter increases the convergence rate of the power iterations, but decreases the convergence rate of the inner iterations. In the performed LWR calculations, an optimal choice of the shift parameter was in the interval [0.02, 0.1] . Applying the power method to the shifted problem (5.4) we get the Wieland method (Nakamura, 1977), also called the method of fractional iterations (Wachspress, 1966). An algorithm of the Wieland method is given in Fig. 5.4.

The convergence of the Wieland method can be further accelerated by applying Chebyshev polynomials. Application of Chebyshev polynomials for acceleration of the power method is discussed in Section 11.2 of (Hageman and Young, 1981). An extension of the method for the shifted eigenvalue problem (5.4) is straightforward. Let us order the eigenvalues of the matrix $\left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]^{-1}\mathbf{F}$ as

choose initial guess $\Phi^{(0)} > 0$ *and* $\lambda^{(0)} > 0$
choose shift parameter $\delta\lambda$
choose number of outer iterations per nonlinear N_Outer
DO $n = 1, 2, \ldots, N\_Outer$
$\lambda' = \lambda^{(n-1)} + \delta\lambda$
solve : $\left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]\Phi^{(n)} = \frac{1}{\Lambda^{(n-1)}}\mathbf{F}\,\Phi^{(n-1)}$
$\Lambda^{(n)} = \Lambda^{(n-1)} \frac{\left\langle \mathbf{e}, \mathbf{F}\,\Phi^{(n)} \right\rangle}{\left\langle \mathbf{e}, \mathbf{F}\,\Phi^{(n-1)} \right\rangle}$
$\lambda^{(n)} = \frac{\Lambda^{(n)}\lambda'}{\Lambda^{(n)} + \lambda'}$
IF *convergence* EXIT
END DO

Figure 5.4: Algorithm of the Wieland method for generalized eigenvalue problem

$$\Lambda_1 > |\Lambda_2| \geq |\Lambda_3| \geq \cdots \geq |\Lambda_{KG}| \,.$$

To apply the Chebyshev acceleration we use the following assumptions

1. the eigenvalues Λ_i are real;
2. the estimate of the dominance ratio $d' = \frac{\Lambda_2}{\Lambda_1}$ and the ratio of the minimum to maximum eigenvalues $b = \frac{\Lambda_{KG}}{\Lambda}$ are available.

In the practical LWR calculations, d' may be chosen from the interval $[0.9, 0.98]$, while $b = 0$. An adaptive procedure to estimate the dominance ratio has been developed based on a comparison of the observed and theoretical convergence rate (Hageman and Young, 1981). However, the adaptive procedure is not reliable in the steady-state calculations, because the both matrices $\mathbf{M}$ and $\mathbf{F}$ are changed at the nonlinear iterations. Algorithm of the Chebyshev acceleration for the power method is taken from the Section 11.2 of (Hageman and Young, 1981). We introduce only two modifications:

1. Chebyshev iterations are started when several power iterations are performed. It increases stability of the algorithm when the matrices $\mathbf{M}$ and $\mathbf{F}$ are changed significantly in the top nonlinear iteration level.

2. Recalculation of the eigenvalue shift λ' is moved into the top nonlinear iteration level. As a result the matrix $\left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]$ does not change during the outer iterations and the Chebyshev acceleration procedure works efficiently.

An algorithm of the Chebyshev acceleration procedure is given in Fig. 5.5.

The algorithm 5.5 is the final algorithm used in the code. Setting $N\_Cheb\_Start > N\_Outer$ we get the Wieland method, because the Chebyshev iterations are not started; if additionally $\frac{1}{\lambda'} = 0$ we obtain the simple iterations of the power method.

The following convergence tests are applied

$$\text{convergence} : \frac{\max_k \left|\Phi_k^{(n)} - \Phi_k^{(n-1)}\right|}{\max_k \left|\Phi_k^{(n)}\right|} \leq \epsilon_\Phi \text{ and } \left|\frac{k_{eff}^{\max} - k_{eff}^{\min}}{2}\right| \leq \epsilon_\lambda$$

where ϵ_Φ is the tolerance of the neutron flux; ϵ_λ is the tolerance of the eigenvalue;

$$k_{eff}^{\max} = \max_k \left|\frac{(\mathbf{F}\Phi)_k^{(n)}}{(\mathbf{F}\Phi)_k^{(n-1)}}\right|; \quad k_{eff}^{\min} = \min_k \left|\frac{(\mathbf{F}\Phi)_k^{(n)}}{(\mathbf{F}\Phi)_k^{(n-1)}}\right|$$

The default values of the both tolerances ϵ_Φ and ϵ_λ are 10^{-5}.

choose initial guess $\Phi^{(0)} > 0$ *and* $\lambda^{(0)} > 0$

choose shift parameter $\delta\lambda$

choose dominace ratio d' *and ratio* b

chose iteration number to start Chebyshev

iterations N_Cheb_Start

choose number of outer iterations

per nonlinear N_Outer

$$p = 0; \quad \gamma = \frac{2}{(2 - d' - b)}; \quad \sigma = \frac{d' - b}{2 - d' - b}$$

$$\lambda' = \lambda^{(0)} + \delta\lambda$$

DO $n = 1, 2, \ldots, N\_Outer$

$$solve: \left[\mathbf{M} - \frac{1}{\lambda'} \mathbf{F} \right] \Phi^{(n)} = \frac{1}{\Lambda^{(n-1)}} \mathbf{F}\, \Phi^{(n-1)}$$

IF $n \geq N\_Cheb\_Start$ THEN

IF $p == 0$ THEN

$$\rho = 1.0$$

ELSE IF $p == 1$ THEN

$$\rho = 1 \Big/ \left(1 - \frac{1}{2} \sigma^2 \right)$$

ELSE

$$\rho = 1 \Big/ \left(1 - \frac{1}{4} \sigma^2 \rho \right)$$

END IF

$$p = p + 1$$

$$\Phi^{(n)} = \rho \left(\gamma\, \Phi^{(n)} + (1 - \gamma)\, \Phi^{(n-1)} \right) + (1 - \rho)\, \Phi^{(n-2)}$$

END IF

$$\Lambda^{(n)} = \Lambda^{(n-1)} \frac{\left\langle \mathbf{e}, \mathbf{F}\, \Phi^{(n)} \right\rangle}{\left\langle \mathbf{e}, \mathbf{F}\, \Phi^{(n-1)} \right\rangle}$$

$$\lambda^{(n)} = \frac{\Lambda^{(n)} \lambda'}{\Lambda^{(n)} + \lambda'}$$

IF *convergence* EXIT

END DO

Figure 5.5: Algorithm of the Chebyshev acceleration procedure applied to the Wieland method for generalized eigenvalue problem

5.3 Inner Iterations

At each outer iterations we should solve the linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{5.5}$$

where $\mathbf{A} \equiv \left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]; \mathbf{x} \equiv \Phi^{(n)}; \mathbf{b} = \frac{1}{\Lambda^{(n-1)}}\mathbf{F}\Phi^{(n-1)}$.

The matrix $\mathbf{A}$ is large and sparse, iterative methods are used for solving Eq. (5.5) forming inner iterations. An application of the iterative method is further motivated by the fact that we do not need to solve Eq. (5.5) exactly, because the right hand side is known only approximately in the beginner of outer iterations. Several iteration strategies are possible to minimise total computing time of the outer and inner iterations, for detail discussion see (Hageman and Young, 1981; Marchuk and Lebedev, 1981; Ferguson and Derstine, 1977). In the SKETCH-N code, we apply a fixed number of inner iterations per outer. In the performed calculations, two inner iterations per outer iteration result in the minimum computing time.

The matrix $\mathbf{A}$ has a block structure with K blocks of the size $G \times G$, where G is a number of neutron energy groups; K is a total number of nodes. The number of energy groups is small in LWR calculations and the matrix blocks are easy to invert and multiply. Thus, the block iterative methods may be efficient for solving Eq. (5.5). In the code, we apply the block symmetric Gauss-Seidel method (Hageman and Young, 1981; Varga, 1962). Let us start with the block Gauss-Seidel method, the i-th iteration of the method can be written as

$$DO\, k = 1, 2, \ldots, K$$

$$\mathbf{x}_k^{(i)} = \mathbf{A}_{kk}^{-1}\left(\mathbf{b}_k - \sum_{j<k}\mathbf{A}_{kj}\mathbf{x}_j^{(i)} - \sum_{j>k}\mathbf{A}_{kj}\mathbf{x}_j^{(i-1)}\right) \tag{5.6}$$

$$ENDDO$$

where

k is the node index;

$\mathbf{x}_k = \mathrm{col}\{x_k^1, x_k^2, \ldots, x_k^G\}$;

$\mathbf{b}_k = \mathrm{col}\{b_k^1, b_k^2, \ldots, b_k^G\}$;

$\mathbf{A}_{kk}$ is the diagonal block of the matrix $\mathbf{A}$;

$\mathbf{A}_{kj}$ is the off-diagonal block located below the block diagonal if $j < k$ and above the block diagonal if $j > k$.

The block symmetric Gauss-Seidel method combines the forward sweep of the Gauss-Seidel method (5.6) and a backward sweep updating the unknowns in reverse order as

$$DO\, k = K, K-1, \ldots, 1$$

$$\mathbf{x}_k^{(i)} = \mathbf{A}_{kk}^{-1} \left(\mathbf{b}_k - \sum_{j<k} \mathbf{A}_{kj} \mathbf{x}_j^{(i-1)} - \sum_{j>k} \mathbf{A}_{kj} \mathbf{x}_j^{(i)} \right)$$

$$ENDDO$$

An algorithm of the symmetric Gauss-Seidel method is given in Fig. 5.6.

The flux vector $\Phi^{(n-1)}$ available from the last outer iteration is used as an initial guess for the inner iterations. Because the block diagonal matrix $\mathbf{A}_{kk} \equiv \left[\mathbf{M} - \frac{1}{\lambda'}\mathbf{F}\right]_{kk}$ does not change during outer iterations, the calculation of the block inverse $\mathbf{A}_{kk}^{-1}$ is done at the top nonlinear iteration level.

choose initial guess $\mathbf{x}^{(0)}$
choose number of inner iterations N_Inner
per outer iteration
$\mathbf{x}^{(1/2)} = \mathbf{x}^{(0)}$
compute $\mathbf{A}_{kk}^{-1}$
DO $i = 1,2,\ldots,N\_Inner$
  DO $k = 1,2,\ldots,K$
    $\mathbf{r} = 0$
    DO $j < k$
      $\mathbf{r} = \mathbf{r} + \mathbf{A}_{kj}\mathbf{x}_j^{(i-1/2)}$
    END DO
    DO $j > k$
      $\mathbf{r} = \mathbf{r} + \mathbf{A}_{kj}\mathbf{x}_j^{(i-1)}$
    END DO
    $\mathbf{x}_k^{(i-1/2)} = \mathbf{A}_{kk}^{-1}\left(\mathbf{b}_k - \mathbf{r}\right)$
  END DO
  DO $k = K, K-1,\ldots,1$
    $\mathbf{r} = 0$
    DO $j < k$
      $\mathbf{r} = \mathbf{r} + \mathbf{A}_{kj}\mathbf{x}_j^{(i-1/2)}$
    END DO
    DO $j > k$
      $\mathbf{r} = \mathbf{r} + \mathbf{A}_{kj}\mathbf{x}_j^{(i)}$
    END DO
    $\mathbf{x}_k^{(i)} = \mathbf{A}_{kk}^{-1}\left(\mathbf{b}_k - \mathbf{r}\right)$
  END DO
END DO

Figure 5.6: Algorithm of the block symmetric Gauss-Seidel method used for inner iterations

5.4 Critical Boron Search

Critical boron search is a variant of the eigenvalue problem typically encountered in PWR applications. We need to find a value of the boron concentration, which results in the reactor multiplication factor (eigenvalue) equal to unity. A problem can be written as the nonlinear equation

$$\lambda(c) - 1 = 0 \tag{5.7}$$

where c is the boron concentration; λ is an eigenvalue of the generalized eigenvalue problem (5.1).

The problem is simplified by assumption that the boron concentration is constant over the reactor domain. The eigenvalue λ is practically linear function of the boron concentration and Eq. (5.7) can be easy solved by the Newton method (Ueberhuber, 1995). Eq. (5.7) is converted into standard form of the nonlinear equation as

$$f(x) = 0$$

where $x \equiv c$; $f(x) \equiv \lambda(x) - 1$.

An iteration of the Newton method is written as

$$x^{(m+1)} = x^{(m+1)} - f(x^{(m)})/f'(x^{(m)}), \quad m = 1, 2, \ldots$$

where m is an nonlinear iteration index; $f'(x)$ is the first derivative of the function $f(x)$.

Because the boron concentration can not be negative, we introduce an additional constraint as

$$x^{(m+1)} = \max\left[x^{(m+1)}, 0\right]$$

The derivative $f'(x^{(m+1)})$ is computed numerically as

$$f'(x^{(m)}) = \frac{f(x^{(m)}) - f(x^{(m-1)})}{x^{(m)} - x^{(m-1)}}, \quad m = 1, 2, \ldots$$

The first step of the method is different because we do not know the function

derivative in advance. At the first step the value of the boron concentration is computed as

$$x^{(1)} = \begin{cases} x^{(0)} + \Delta x^{(0)} & \text{if } f(x^{(0)}) > 0 \\ \max[x^{(0)} - \Delta x^{(0)}, 0] & \text{if } f(x^{(0)}) < 0 \end{cases}$$

where $\Delta x^{(0)}$ is the initial change of the boron concentration, the value 200 ppm is used in the code.

The critical boron search is incorporated into the top level of the nonlinear iterations. We start the search if the eigenvalue is computed within a tolerance 0.01. The boron search terminates if the eigenvalue is equal to unity within a given tolerance, the value $10^{-5} = 1\,pcm$ is the default value in the code. Due to a linear dependence of the eigenvalue on the boron concentration, only a few iterations of the Newton method are usually needed.

Chapter 6

Iterative solution of neutron kinetics problems

In the Chapter 4, we derived the coarse-mesh finite-difference form of the neutron kinetics equations. Iterative techniques to solve these equations are presented in this Chapter.

6.1 Global Iteration Strategy

The coarse-mesh finite-difference equations (4.7) are written in the matrix form as

$$\mathbf{A}[\Phi(t+\Delta t), \mathbf{T}(t+\Delta t)]\, \Phi(t+\Delta t) = \mathbf{b}\,, \tag{6.1}$$

where

$\Phi(t+\Delta t) = col\left\{(\overline{\Phi}^1(t+\Delta t), \ldots, \overline{\Phi}^K(t+\Delta t)\right\}$ is the vector of the node-averaged neutron flux at the time moment $t+\Delta t$; size of the vector is KG, K is the total number of the nodes, G is the number of neutron energy groups;

$\mathbf{A}[\Phi(t+\Delta t), \mathbf{T}(t+\Delta t)]$ is the $KG \times KG$ matrix, containing the finite-difference entries of the diffusion, removal, scattering and fission operators; the dependency of the matrix $\mathbf{A}$ on the neutron flux Φ and the thermal-hydraulics parameters $\mathbf{T}$ is explicitly stated.

$\mathbf{b}$ is the right hand side vector of size KG.

In the transient calculations, the neutron flux and thermal-hydraulics parameters do not change significantly at the time step. Thus, we linearized Eq. (6.1) using the neutron flux and the thermal-hydraulics parameters from the previous time step to compute the matrix $\mathbf{A}$. As a result we do not perform nonlinear iterations during transient calculations, we need only to solve the linear system

$$\mathbf{A}\,\Phi(t+\Delta t) = \mathbf{b}\,, \tag{6.2}$$

where $\mathbf{A} \equiv \mathbf{A}[\Phi(t), \mathbf{T}(t)]$.

6.2 Basic Iterative Method

The block symmetric Gauss-Seidel (BSGS) method described in the Section 5.3 can be used for solving Eq. (6.2). The i-th iteration of the BSGS method can be written as

$$\mathbf{M}\Phi^{(i)} = (\mathbf{M}-\mathbf{A})\Phi^{(i-1)} + \mathbf{b}, \quad i = 1, 2, \ldots, \tag{6.3}$$

where $\mathbf{M}$ is called the preconditioner matrix.

In the BSGS method, the matrix $\mathbf{M}$ is given as

$$\mathbf{M} = (\mathbf{D}_A - \mathbf{L}_A)\mathbf{D}_A^{-1}(\mathbf{D}_A - \mathbf{U}_A),$$

where

$\mathbf{D}_A$ is the block diagonal part of the matrix $\mathbf{A}$;

$\mathbf{L}_A$ is the block lower triangular part of the matrix $\mathbf{A}$;

$\mathbf{U}_A$ is the block upper triangular part of $\mathbf{A}$.

The iteration (6.3) can be written in the form

$$\Phi^{(i)} = \mathbf{G}\Phi^{(i-1)} + \mathbf{f}, \quad i = 1, 2, \ldots, \tag{6.4}$$

where

$\mathbf{G} \equiv \mathbf{M}^{-1}(\mathbf{M} - \mathbf{A}) = \mathbf{I} - \mathbf{M}^{-1}\mathbf{A}$ is called the iteration matrix;

$\mathbf{f} \equiv \mathbf{M}^{-1}\mathbf{b}$.

If a residual of the iterative process (6.4) is defined as

$$\delta^{(i)} = \mathbf{G}\Phi^{(i-1)} + \mathbf{f} - \Phi^{(i-1)}, \quad i = 1, 2, \ldots$$

The iterations of the BSGS method are given by the algorithm in the Fig. 6.1.

choose initial guess $\Phi^{(0)}$
DO $i = 1, 2, \ldots,$
$\boldsymbol{\delta}^{(i)} = \mathbf{G}\,\Phi^{(i-1)} + \mathbf{f} - \Phi^{(i-1)}$
$\Phi^{(i)} = \Phi^{(i-1)} + \boldsymbol{\delta}^{(i)}$
IF *convergence* EXIT
END DO

Figure 6.1: Algorithm of the basic iterative method

Please, note that the matrix $\mathbf{G}$ is not actually used in the calculations, we simply perform several iterations of the BSGS method using the algorithm given in Fig. 5.6 to compute the approximation $\mathbf{G}\Phi^{i-1} + \mathbf{f}$. A convergence rate of the method is determined by the spectral radius (maximum eigenvalue) of the iteration matrix $\mathbf{G}$. In practical LWR calculations, the spectral radius is close to unity and convergence of the BSGS method is slow. The following section describes the acceleration of the basic iterative method using Chebyshev polynomials.

6.3 Adaptive Chebyshev Acceleration Procedure

The basic iterative method given in Fig. 6.1 can be accelerated using the Chebyshev semi-iterative method (Hageman and Young, 1981; Varga, 1962). We make an assumption that the eigenvalues λ_i of the matrix $\mathbf{G}$ are real and can be ordered as

$$m(\mathbf{G}) \equiv \lambda_{KG} \leq \cdots \leq \lambda_1 \equiv M(\mathbf{G})$$

To apply the Chebyshev polynomials we need the estimates m_E and M_E of the minimum and maximum eigenvalues $m(\mathbf{G})$ and $M(\mathbf{G})$. In the case of the BSGS method, the estimate of the minimum eigenvalue can be taken as zero, the estimate of the maximum eigenvalue is computed using the adaptive procedure comparing the theoretical and observed convergence rates. In the following, we skip theoretical details of the both Chebyshev semi-iterative method and the adaptive procedure; they are given in the book of Hageman and Young (1981). The algorithm taken from the same book is presented in Fig. 6.2; the calculation of the Chebyshev iterative parameters is shown in Fig. 6.3, calculation of the estimate of the maximum eigenvalue is given in Fig. 6.4.

The input quantities used in the Chebyshev acceleration procedure are:

$\Phi^{(0)}$ is the initial guess vector;

tol is tolerance criteria;

m_E is the lower bound for $m(\mathbf{G})$, the minimum eigenvalue of the iteration matrix $\mathbf{G}$; $m_E = 0$ in the case of the BSGS method;

M_E is the initial estimate for $M(\mathbf{G})$, the maximum eigenvalue of the iteration matrix $\mathbf{G}$; M_E must satisfy $M_E < M(\mathbf{G}) < 1$; $M_E = 0.8$ is the default value in the code;

τ is the upper bound for the maximum eigenvalue estimate; if $M'_E > \tau$ then $M'_E = \tau$;

$\tau = 0.995$ is the default value;

F is the damping factor used in the parameter change test; if $F = 0$ the estimate M_E is not changed and we have nonadaptive procedure; F may be chosen from the interval$[0.65, 0.85]$; $F = 0.8$ is the default value;

N_Cheb_Start is the iteration number when the Chebyshev iterations start; $N\_Cheb\_Start = 5$ is the default value;

$P\_Estimate\_Change$ is the minimum degree of the Chebyshev polynomial when the estimate M_E can be changed; $P\_Estimate\_Change = 5$ is the default value.

Input :

$\Phi^{(0)}, tol, M_E, m_E, \tau, F, N\_Cheb\_Start, P\_Estimate\_Change$

Initialize :

$n = 0;\ p = 0;\ s = 0;\ M'_E = M_E;\ r\_norm = 1.0;$

DO $n = 1, 2, \ldots,$

$r\_norm\_p = r\_norm;$

IF $n \geq N\_Cheb\_Start$ *THEN*

compute Chebyshev parameters ρ *and* γ :

see Fig.6.3

$p = p + 1$

ELSE

$\rho = 1.0; \quad \gamma = 1.0$

END IF

calculate new iterate :

$$\boldsymbol{\delta}^{(n)} = \mathbf{G}\,\Phi^{(n-1)} + \mathbf{f} - \Phi^{(n-1)}; \quad r\_norm = \left\|\boldsymbol{\delta}^{(n)}\right\|_2$$

$$\Phi^{(n)} = \rho\left(\gamma\,\delta^{(n-1)} + \Phi^{(n-1)}\right) + (1-\rho)\,\Phi^{(n-2)}; \; f\_norm = \left\|\Phi^{(n)}\right\|_2$$

calculate new estimate M'_E :

see Fig.6.4

parameter change test :

IF $p \geq P\_Estimate\_Change$ *.AND.* $B > (Q)^F$ *THEN*

$p = 0$

END IF

converergence test :

$$\text{IF}\left(\frac{1}{1-M'_E}\right)\frac{r\_norm}{f\_norm} \leq tol \text{ EXIT}$$

END DO

Figure 6.2: Algorithm of the adaptive Chebyshev acceleration procedure applied to the basic iterative method

The iteration counters used in the algorithm are

n is the current iteration number;

p is the degree of the Chebyshev polynomial;

s is the number of the Chebyshev polynomial sequence.

IF $p == 0$ *THEN*

$s = s + 1$ $<$ *new Chebyshev polynomial sequaence* $>$

IF $M'_E > \tau$ *THEN*

$M'_E = \tau$

END IF

$M_E = M'_E$

$$\gamma = \frac{2}{2 - M_E - m_E}; \quad \sigma_E = \frac{M_E - m_E}{2 - M_E - m_E};$$

$$\rho = 1.0; \quad r = \frac{1 - \sqrt{1 - \sigma_E^2}}{1 + \sqrt{1 - \sigma_E^2}}$$

ELSE IF $p == 1$ *THEN*

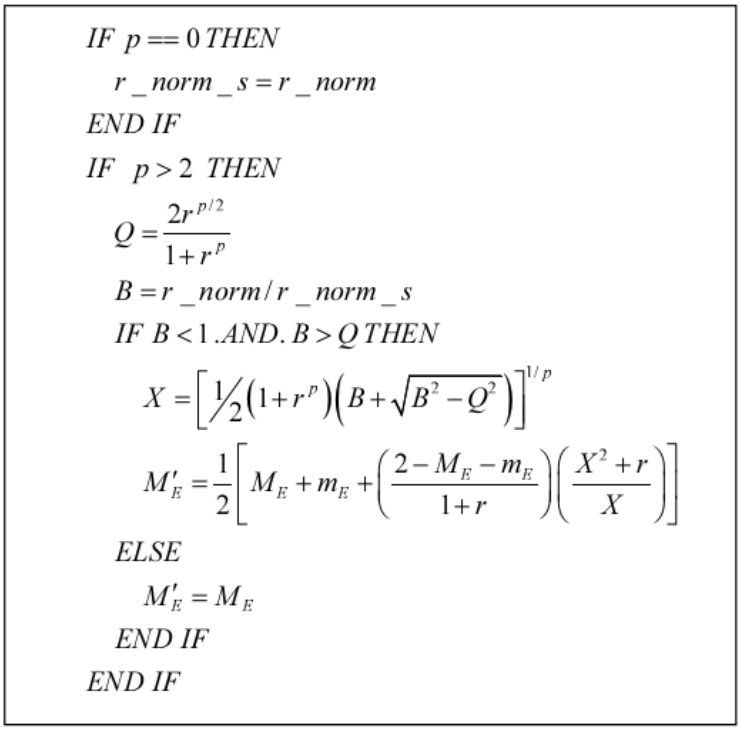
$$\rho = 1 \Big/ \left(1 - \frac{1}{2}\sigma_E^2\right)$$

ELSE

$$\rho = 1 \Big/ \left(1 - \frac{1}{4}\sigma_E^2 \rho\right)$$

END IF

Figure 6.3: Algorithm to compute the Chebyshev iterative parameters ρ and γ



$$
\begin{aligned}
&IF\ p == 0\ THEN\\
&\quad r\_norm\_s = r\_norm\\
&END\ IF\\
&IF\ \ p > 2\ \ THEN\\
&\quad Q = \frac{2r^{p/2}}{1+r^{p}}\\
&\quad B = r\_norm / r\_norm\_s\\
&\quad IF\ B < 1\ .AND.\ B > Q\ THEN\\
&\qquad X = \left[\tfrac{1}{2}\left(1+r^{p}\right)\left(B+\sqrt{B^{2}-Q^{2}}\right)\right]^{1/p}\\
&\qquad M'_{E} = \frac{1}{2}\left[M_{E} + m_{E} + \left(\frac{2-M_{E}-m_{E}}{1+r}\right)\left(\frac{X^{2}+r}{X}\right)\right]\\
&\quad ELSE\\
&\qquad M'_{E} = M_{E}\\
&\quad END\ IF\\
&END\ IF
\end{aligned}
$$

Figure 6.4: Algorithm to compute the estimation M'_E of the maximum eigenvalue of the iteration matrix

Chapter 7

Macro cross section model

In the neutron diffusion calculations, we assume that the reactor domain is subdivided into the nodes and the macro cross sections are constant over the node. One or four nodes per fuel assembly are typically used in LWR calculations. Based on the material composition the nodes are grouped into the node types and for each type the macro cross sections are computed in advance using the transport theory codes like WIMS, CASMO, HELIOS, etc. The macro cross sections depend on the feedback variables such as burnup, Doppler fuel temperature, coolant void fraction, etc., and the transport calculations are performed to cover a range of the change of the feedback variables. As a result we have a multidimensional table of the macro cross sections depending on the feedback variables. In the neutron diffusion code, the macro cross sections can be computed by interpolation in this table, however due to computer memory limitation, the table dimension is usually reduced approximating the macro cross section by some functions. Macro cross section model of the neutron diffusion code assumes a certain form of the dependencies of the macro cross sections on the feedback variables. Two macro cross section models are available in the SKETCH-N code. The first model, developed for the calculations of the NEACRP LWR core transient benchmark (Finnemann and Galati, 1992) is based on the representation of the macro cross sections as polynomial functions of boron concentration, Doppler fuel temperature, temperature and density of coolant. The model is described in Section 7.1. The second model is developed for the data provided in NEA/NSC Ringhals-1 BWR

stability benchmark (Lefvert, 1994). In this model, the macro cross sections are given as a combination of a three-dimensional table with entries for burnup, void and conversion history and additional polynomial dependencies for the Doppler feedback and Xenon poisoning. The model is presented in Section 7.2. The both macro cross section models were developed for the given macro cross section data and they are not general enough to cover all LWR transients. An application of the code to other transients or other reactor types will require a development of a new model. The macro cross sections depend also on the position of a control rod. The control rods are moving during reactor transients and we have a problem to compute the macro cross sections of the nodes with a partially inserted control rod. Traditional volume-weighting homogenization procedure gives significant errors, known as a rod cusping effect. The effect is especially important in the case of PWR rod ejection and rod withdrawal transients. Flux-weighting homogenization procedure, which significantly reduces these errors, is described in the Section 7.3.

7.1 Polynomial Model

Two neutron energy groups are used in the LWR core transient benchmark (Finnemann and Galati, 1992). Neutron velocities and energy release per fission are considered to be independent in time and space. Six groups of delayed neutron precursors are applied. Time constants and fractions of the delayed neutrons are constants over the reactor domain. Within the reactor domain different material compositions and corresponding sets of the macro cross sections are defined. For each material composition the following macro cross sections are given

Σ_{r1}, Σ_{r2} are the transport macro cross sections;

Σ_{a1}, Σ_{a2} are the absorption macro cross sections;

Σ_{s21} is the scattering macro cross section from the group 1 into group 2;

Σ_{f1}, Σ_{f2} are the fission macro cross sections;

$\nu\Sigma_{f1}, \nu\Sigma_{f2}$ are the multiplication macro cross sections.

The diffusion coefficient is computed in terms of the transport macro cross

section as

$$D_g = 1/(3\Sigma_{tr,g}), \quad g = 1, 2$$

The removal macro cross sections are computed as

$$\Sigma_{r1} = \Sigma_{a1} + \Sigma_{s21}, \quad \Sigma_{r2} = \Sigma_{a2}.$$

The macro cross sections depend on the following feedback variables

c is the borom concentration [ppm];

ρ is the coolant density [g/cm^3];

T_M is the coolant temperature [°C];

T_D is the Doppler fuel temperature [°K].

The Doppler fuel temperature is computed interpolating the fuel rod centerline temperature $T_{F,C}$ and the fuel rod surface temperature $T_{F,S}$ as

$$T_D = (1 - \alpha)T_{F,C} + \alpha T_{F,S},$$

where α is taken equal 0.7.

The macro cross sections are expressed in terms of the feedback variables as

$$\Sigma = \Sigma_0 + (\partial\Sigma/\partial c)_0(c - c_0) + (\partial\Sigma/\partial\rho)_0(\rho - \rho_0) + (\partial\Sigma/\partial T_M)_0(T_M - T_{M0}) + (\partial\Sigma/\partial\sqrt{T_D})_0(\sqrt{T_D} - \sqrt{T_{D0}}),$$

where index 0 marks the reference values.

The reference macro cross sections, their derivatives and the feedback reference values are given for each material composition.

The macro cross section of the node with a control rod is determined by adding the differential cross section $\Delta\Sigma^{CR}$ to the cross section without control rod as

$$\Sigma^{\text{with CR}} = \Sigma^{\text{without CR}} + \Delta\Sigma^{CR}.$$

The differential cross section $\Delta\Sigma$ does not depend on the feedbacks. The contribution of the control rod driver is treated in the same way. The treatment

of the node with partially inserted control rod will be given in Section 7.3

7.2 Model for Ringhals-1 BWR Stability Benchmark

Ringhals-1 BWR stability benchmark has been recently performed by Nuclear Science Committee of OECD Nuclear Energy Agency for validation of computer codes used in BWR stability analysis. The data come from measurements performed in Swedish reactor Ringhals-1 designed by ABB Atom and owned and operated by Vattenfall AB. An extensive set of the data describing Ringhals-1 reactor is given in the benchmark specification report (Lefvert, 1994), Ringhals report (Johansson, 1994) and the files on CD-ROM. Two neutron energy groups and six groups of delayed neutrons are used. Within reactor domain different material compositions and the corresponding sets of the macro cross sections are defined. For each material composition the following macro cross sections are given

D_1, D_2 are the diffusion coefficients;

Σ_{a1}, Σ_{a2} are the absorption macro cross sections;

Σ_{s21} is the scattering macro cross section from the group 1 into group 2;

$\nu\Sigma_{f1}, \nu\Sigma_{f2}$ are the multiplication macro cross sections;

ν is the neutron yield per fission.

For each material composition the macro cross sections are given as three-dimensional tables with entries for burnup, void and conversion history. Conversion history is a collective variable taking into account void history and control rod history. To compute the macro cross section for intermediate values of the void, burnup and conversion history, a three-dimensional tensor-product linear interpolation (Press et al., 1992; Ueberhuber, 1995) is applied. Let us mathematically define the problem. Feedback variables form a three-dimensional Cartesian grid as x_l, $l = 1, \ldots, L$; y_m, $m = 1, \ldots, M$; z_n, $n = 1, \ldots, N$; and the macro cross sections are the functions given at the grid points as $f_{lmn} = f(x_l, y_m, z_n)$. We need to find a value of the function f at the point (x, y, z), where

$x \in [x_l, x_{l+1}]$; $y \in [y_m, y_{m+1}]$ and $z \in [z_n, z_{n+1}]$. The value $f(x, y, z)$ is computed using the tensor-product linear interpolation (Ueberhuber, 1995), which is defined recursively as

$$f^{1D}(\xi, f_l, f_{l+1}) = (1-\xi) f_l + \xi f_{l+1};$$

$$f^{2D}[\xi, \eta, f_{l,m}, f_{l+1,m}, f_{l,m+1}, f_{l+1,m+1}] = (1-\eta) f^{1D}[\xi, f_{l,m}, f_{l+1,m}] + \eta f^{1D}[\xi, f_{l,m+1}, f_{l+1,m+1}];$$

$$\begin{aligned} f^{3D}[\xi, \eta, \zeta, f_{l,m,n}, f_{l+1,m,n}, f_{l,m+1,n}, f_{l+1,m+1,n}, f_{l,m,n+1}, f_{l+1,m,n+1}, f_{l,m+1,n+1}, f_{l+1,m+1,n+1}] = \\ (1-\zeta) f^{2D}[\xi, \eta, f_{l,m,n}, f_{l+1,m,n}, f_{l,m+1,n}, f_{l+1,m+1,n}] + \\ \zeta f^{2D}[\xi, \eta, f_{l,m,n+1}, f_{l+1,m,n+1}, f_{l,m+1,n+1}, f_{l+1,m+1,n+1}]; \end{aligned} \quad (7.1)$$

where

$$\xi = \frac{x - x_l}{x_{l+1} - x_l}; \quad \eta = \frac{y - y_m}{y_{m+1} - y}; \quad \zeta = \frac{z - z_n}{z_{n+1} - z_n};$$

and f^{1D}, f^{2D}, f^{3D} are one-, two-, and three-dimensional linear interpolation, respectively.

The macro cross section of the node with a control rod is determined by adding the differential cross section $\Delta\Sigma^{\mathrm{CR}}$ contributed by a control rod to the cross section without control rod as

$$\Sigma^{\text{with CR}} = \Sigma^{\text{without CR}} + \Delta\Sigma^{\mathrm{CR}}$$

The differential cross sections for absorption cross section $\Delta\Sigma_{a2}^{\mathrm{CR}}$ and for multiplication cross section $\Delta\nu\Sigma_{a2}^{\mathrm{CR}}$ are given also as three-dimensional table with entries for burnup, void and conversion history for each material composition. Their values are computed by the three-dimensional linear interpolation (7.1). The other differential cross sections are constant for each material composition. The data for the Doppler feedback are given as a change of the infinite multiplication factor as

$$\Delta k_{\infty}^{Doppler} = D(E, \nu) \left(\sqrt{T_{fuel}} - \sqrt{T_0} \right), \quad (7.2)$$

$$D(E,\, v) = (1 + d_1 \times E + d_2 \times E^2 + d_3 \times E^3) \times (d_4 + d_5 \times v),$$

where

$\Delta k_\infty^{Doppler}$ is the change of the infinite multiplication factor due to the Doppler effect;

T_{fuel} is the Doppler fuel temperature;

T_0 is the nominal temperature at which the cross sections have been generated (791 K);

E is the burnup,

v is the instantaneous void,

$d_1, \ldots, d_5$ are the coefficients given for each fuel type.

In the case of two neutron energy groups, the infinite multiplication factor k_∞^0 is computed as

$$k_\infty^0 = \frac{\nu\Sigma_{f1}^0 + \frac{\Sigma_r^0}{\Sigma_{a2}^0}\nu\Sigma_{f2}^0}{\Sigma_r^0 + \Sigma_{a1}^0},$$

where index 0 marks the values computed at the reference Doppler fuel temperature 791 K. The Doppler effect is incorporated into the fast group absorption macro cross section estimated from the condition to preserve the change of the infinite multiplication factor given by Eq. (7.2). The fast group absorption macro cross section is computed as

$$\Sigma_{a1}^{Doppler} = \frac{\nu\Sigma_{f1}^0 + \frac{\Sigma_r^0}{\Sigma_{a2}^0}\nu\Sigma_{f2}^0}{\Delta k_\infty^{Doppl} + k_\infty^0} - \Sigma_r^0.$$

In the benchmark, the microscopic xenon thermal absorption cross section σ_{a2}^{xe} is specified as

$$\sigma_{a2}^{xe} = (s_1 + s_2 \times E) \times (s_3 + s_4 \times v + s_4 \times v^2),$$

where $s_1, \ldots, s_5$ are the coefficient given for each material composition.

In the code, Xenon poisoning is taking into account as

$$\Sigma_{a2}^{xe} = \Sigma_{a2}^{0} + \sigma_{a2}^{xe} \times (Xe - Xe^{0}),$$

where index 0 marks macro cross sections computed with the nominal xenon concentration Xe^0. The reference xenon concentration Xe^0 is not specified in benchmark data. The only reference about its value is given in (Johansson, 1994), where is written "*Absorption cross section for thermal neutrons SIGA2 include xenon corrections (i.e. they are calculated assuming equilibrium xenon at full power)*".

We compute approximately an equilibrium xenon concentration as follows

$$Xe^{0} = \frac{(\gamma_1 + \gamma_{xe})\Sigma_f \Phi_2^{Ref}}{\lambda_{xe} + \sigma_{xe}\Phi_2^{Ref}},$$

where

γ_1 is the effective yield of iodine (0.064 atoms/fission);

γ_{xe} is the effective yield of xenon (0.003 atoms/fission);

λ_{xe} is the xenon decay constant (2.09E-05 s^{-1});

σ_{xe} is the microscopic xenon absorption cross section;

Φ_2^{Ref} is the reference thermal flux;

Σ_f is the effective fission cross section computed as $\Sigma_f = \Sigma_{f2} + \Sigma_{f1}\Phi_1/\Phi_2 \approx \Sigma_{f2} + \Sigma_{f1}\Sigma_{a2}/\Sigma_R$.

Xenon constants are taken from (Cameron, 1982). The reference thermal flux for the specified reactor power is computed as

$$\Phi_2^{Ref} = \frac{P_{NOM}}{V_{core}\varepsilon\Sigma_f},$$

where

P_{NOM} is the nominal reactor power (2270 MWt);

V_{core} is the volume of the reactor core (5.5640E+7 cm^3);

ε is the conversion factor (3.237E-11 W/fission).

7.3 Treatmest of the Node with Partially Isserted Costrol Rod

In the nodal calculations, we assume that the macro cross sections are constant over the node and two sets of the data with and without control rods are usually given. In the actual calculations, the control rod can be in an intermediate position inside a node, thus introducing a problem of the macro cross section homogenization. The simplest method is a volume-weighting homogenization procedure given by the formula

$$\Sigma^{l,m,n} = (1-\chi)\Sigma^{\text{without CR}} + \chi\Sigma^{\text{with CR}}, \tag{7.3}$$

where

χ is the weighting factor computed using the volume-weighting as $\chi = \frac{h_n^{\text{with CR}}}{h_n^{\text{with CR}} + h_n^{\text{without CR}}}$;

$h_n^{\text{with CR}}$ is the height of the rodded part of the node;

$h_n^{\text{without CR}}$ is the height of the unrodded part of the node.

The neutron flux changes sharply at the interface of the rodded and unrodded parts as illustrated in Fig. 7.1.

As a result, the volume-weighting homogenization procedure (7.3) introduces significant errors in PWR calculations, known as a rod cusping effect. The term "rod cusping" is due to the cusp-like behavior of the node-averaged flux versus time when the control rod moves through the node. These errors can be reduced applying the flux-volume homogenization. The macro cross sections are computed using the same formula (7.1), but the weighting factor χ is calculated using flux-weighting i.e.

$$\chi_g = \frac{\Phi_g^{\text{with CR}} h_n^{\text{with CR}}}{\Phi_g^{\text{with CR}} h_n^{\text{with CR}} + \Phi_g^{\text{without CR}} h_n^{\text{without CR}}}$$

where

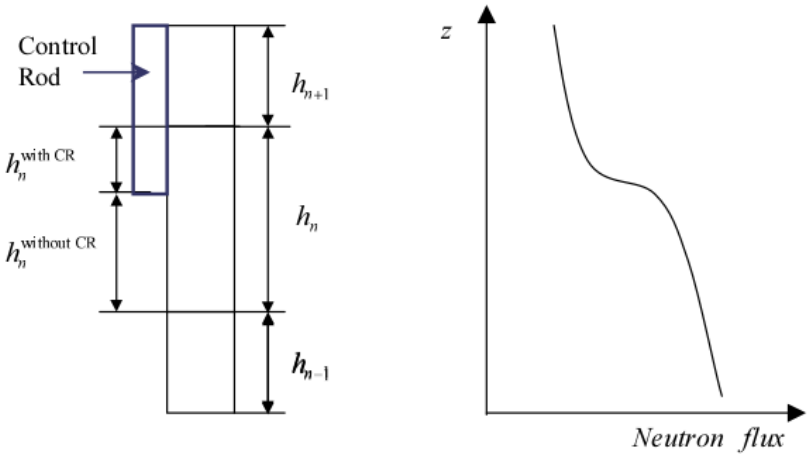


Figure 7.1: Illustration of the rod cusping effect

g is an index of neutron energy group;

$\Phi_g^{\text{with CR}}$ is the average flux of the rodded part of the node;

$\Phi_g^{\text{without CR}}$ is the average flux of the unrodded part of the node.

These two values $\Phi_g^{\text{with CR}}$ and $\Phi_g^{\text{without CR}}$ are not known from the results of the nodal calculations. In the case of a strongly absorbing control rod, which is the case when the rod cusping effect is significant, the neutron flux changes sharply at the rod tip and varies slowly away from the tip as is shown in Fig. 7.1. Based on this observation a simple approximating procedure to compute $\Phi_g^{\text{with CR}}$ and $\Phi_g^{\text{without CR}}$ has been proposed (Gehin, 1992). The flux in the unrodded part of the node is calculated by averaging the fluxes of the partially rodded node and its lower neighbor as

$$\Phi_g^{\text{without CR}} = \xi \Phi_g^{l,m,n} + (1-\xi)\Phi_g^{l,m,n-1},$$

where $\xi = \frac{h_n^{\text{without CR}}}{h_n^{\text{without CR}} + h_{n-1}}$.

Likewise, the average flux in the rodded part of the node is approximated by averaging the fluxes in partially rodded node and its upper neighbor as

$$\Phi_g^{\text{with CR}} = \eta \Phi_g^{l,m,n} + (1-\eta)\Phi_g^{l,m,n+1},$$

where $\eta = \frac{h_n^{\text{with CR}}}{h_n^{\text{with CR}} + h_{n+1}}$.

This approach works well in the performed PWR rod ejection calculations, the rod cusping errors are small even in the case of the coarse axial mesh size of 30 cm (Zimin et al., 1999). However, in the case of the PWR rod withdrawal transients, when several control rods are moving, a fine axial mesh is needed to get accurate results (Asaka et al., 2000). More elaborate approach should be developed if a code user wants to perform general PWR calculations with a coarse axial mesh.

Chapter 8

Internal thermal-hydraulics model

Chapter 8 describes an internal thermal-hydraulics model of the SKETCH-N code. The model is developed for PWR calculations with single-phase coolant flow. Heat conduction equations in fuel rods are discretized in time and space using the finite-difference method. Resulting three-diagonal system of equations is solved using LU decomposition. Heat conduction is considered only in radial direction. Material properties are taken from the NEACRP LWR core transient benchmark specification (Finnemann and Galati, 1992). Fluid dynamics is modeled under single-phase flow conditions. The mass flow rate in each coolant channel is assumed to be known and specified by a code user. As a result we need to solve only mass continuity and energy conservation equations. They are discretized in space using the finite-difference method and in time by the semi-implicit scheme. The heat conduction model has been verified by a comparison with an analytical solution available for steady-state conditions. The thermal-hydraulics model has been applied for the NEACRP PWR rod ejection benchmark (Finnemann and Galati, 1992) and demonstrated accurate results for fuel and coolant temperatures.

8.1 Heat Conduction in the Fuel Rod

The heat conduction equation in the fuel rod is written as

$$\rho(r,t)\ C(r,t)\frac{\partial T(r,t)}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r} r\ k(r,t)\frac{\partial}{\partial r}\ T(r,t) + q(r,t) \tag{8.1}$$

where

$T(r,t)$ is the temperature at the point r along the fuel pin radius [°K];

$k(r,t)$ is the thermal conductivity [watts/m °K];

$q(r,t)$ is the volumetric power generation rate [watts/m^3];

$C(r,t)$ is the specific heat of the fuel pin material [J/kg °K];

$\rho(r,t)$ is the fuel pin density [kg/m^3].

Eq. (8.1) is completed by the boundary conditions:

$$-k(r,t)\,\frac{\partial}{\partial r}\,T(r,t)\bigg|_{r=0} = 0 \tag{8.2}$$

$$-k(r,t)\,\frac{\partial}{\partial r}\,T(r,t)\bigg|_{r=R_F} = q''(R_F,t) = \frac{R_{CL}}{R_F} q''(R_{CL}^{in},t) \tag{8.3}$$

$$-k(r,t)\,\frac{\partial}{\partial r}\,T(r,t)\bigg|_{r=R_{CL}^{in}} = q''(R_{CL}^{in},t) = H_g\left[T(R_F,t) - T(R_{CL}^{in},t)\right] \tag{8.4}$$

$$-k(r,t)\,\frac{\partial}{\partial r}\,T(r,t)\bigg|_{r=R_{CL}^{out}} = q''(R_{CL}^{out},t) = h\left[T(R_{CL}^{out},t) - T_L(t)\right] \tag{8.5}$$

where

R_F is the radius of the fuel pellet [m];

R_{CL}^{in} is the inner radius of the cladding [m];

R_{CL}^{out} is the outer radius of the cladding [m];

H_g is the gap conductance [W/m^2 °K];

$q''(R_{CL}^{in},t)$ is the heat flux at the inner surface of the cladding [W/m^2];

$q''(R_F,t)$ is the heat flux at the outer surface of the fuel pellet [W/m^2];

$T_L(t)$ is the bulk coolant temperature [°K].

The heat conduction Eq. (8.1) is discretized at the spatial mesh given in Fig. 8.1, where the solid lines show locations of the mesh points and the dashed lines indicate control volumes. The temperatures are computed at the points $R_{r,m}$, $m = 1, \ldots, M+1$ and $R_{c,m}$, $m = 1, \ldots, M+1$. The control volumes are taken as the intervals $[r_{f,m-1}, r_{f,m}]$ and $[r_{c,m-1}, r_{c,m}]$ for fuel and cladding, respectively. The derivation of the heat conduction equations is performed in the way as described in (Patankar, 1980). Let as consider the inner spatial region $r_{m,1} \leq r \leq r_m$. Eq. (8.1) is multiplied by r and integrated over the interval $[r_{m-1},\ r_m]$. The result is as follows

$$\left[\frac{r_m^2 - r_{m-1}^2}{2}\right] \rho_m(t)\ \ C_m(t)\ \frac{\partial T_m(t)}{\partial t} = r_m k(r_m, t) \frac{\partial}{\partial r}\ T(r,t)\Big|_{r=r_m} -$$

$$r_{m-1} k(r_{m-1}, t) \frac{\partial}{\partial r}\ T(r,t)\Big|_{r=r_{m-1}} + q_m(t) \left[\frac{r_m^2 - r_{m-1}^2}{2}\right]$$

where

$q_m(t)$ is the average volumetric power generation rate at the interval $[r_{m-1}, r_m]$,

$T_i(t)$ is the average temperature at the interval $[r_{m-1}, r_m]$.

The heat flux at the interface between the nodes m and $m+1$ is expressed as follows

$$k(r_m, t)\ \frac{\partial}{\partial r}\ T(r,t)\Big|_{r=r_m} = \frac{2k_m(t)\ k_{m+1}(t)}{k_m(t) + k_{m+1}(t)}\ \frac{(T_{m+1}(t) - T_m(t))}{\Delta R_m}$$

where $\Delta R_m = R_{m+1} - R_m$.

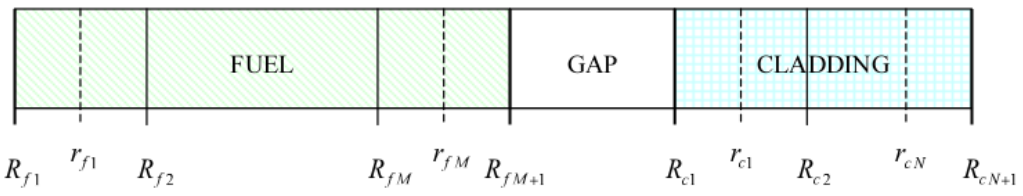


Figure 8.1: Radial spatial mesh in fuel rods

The resulting equation for the internal node is

$$\left[\frac{r_m^2 - r_{m-1}^2}{2}\right] \rho_m(t) C_m(t) \frac{\partial T_m(t)}{\partial t} = r_m \frac{2k_m(t)k_{m+1}(t)}{\Delta R_m(k_m(t) + k_{m+1}(t))}(T_{m+1}(t) - T_m(t)) + r_{m-1} \frac{2k_m(t)k_{m-1}(t)}{\Delta R_{m-1}(k_m(t) + k_{m-1}(t))}(T_{m-1}(t) - T_m(t)) + \left[\frac{r_m^2 - r_{m-1}^2}{2}\right] \times q_m(t) \quad (8.6)$$

Using the boundary conditions (8.2)-(8.5) for the heat flux, the equations are also discretized for the boundary mesh points. The results are as follows:

centerline fuel temperature:

$$\left[\frac{r_{f1}^2}{2}\right] \rho_{f1}(t) C_{f1}(t) \frac{\partial T_{f1}(t)}{\partial t} = r_{f1} \frac{2k_{f2}(t)k_{f1}(t)}{\Delta R_{f1}(k_{f1}(t) + k_{f2}(t))}(T_{f2}(t) - T_{f1}(t)) + \left[\frac{r_{f1}^2}{2}\right] \times q_1(t) \quad (8.7)$$

surface fuel temperature:

$$\left[\frac{R_{fM+1}^2 - r_{fM}^2}{2}\right] \rho_{fM+1}(t) C_{fM+1}(t) \frac{\partial T_{fM+1}(t)}{\partial t} = R_{c1} H_g \left(T_{c1}(t) - T_{fM+1}(t)\right) + r_{fM} \frac{2k_{fM}(t)k_{fM+1}(t)}{\Delta R_{fM}(k_{fM}(t) + k_{fM+1}(t))}(T_{fM}(t) - T_{fM+1}(t)) + \left[\frac{R_{fM+1}^2 - r_{fM}^2}{2}\right] \times q_{M+1}(t) \quad (8.8)$$

inner surface cladding temperature:

$$\left[\frac{r_{c_2}^2 - R_{c1}^2}{2}\right] \rho_{c1}(t) C_{c1}(t) \frac{\partial T_{c1}(t)}{\partial t} = r_{c1} \frac{2k_{c1}(t)k_{c2}(t)}{\Delta R_{c1}(k_{c1}(t) + k_{c2}(t))}(T_{c2}(t) - T_{c1}(t)) + R_{c1} H_g \left(T_{f_M+1}(t) - T_{c1}(t)\right) \quad (8.9)$$

outer surface cladding temperature:

$$\left[\frac{R_{cN+1}^2 - r_{cN}^2}{2}\right] \rho_{cN+1}(t) C_{cN+1}(t) \frac{\partial T_{cN+1}(t)}{\partial t} = R_{cN+1} h \left(T_{bulk}(t) - T_{cN+1}(t)\right) + r_{cN} \frac{2 k_{cN+1}(t) k_{cN}(t)}{\Delta R_{cN} (k_{cN+1}(t) + k_{cN}(t))} (T_{cN}(t) - T_{cN+1}(t)) \quad (8.10)$$

The equations (8.6)-(8.10) are discretized in time using a fully-implicit scheme. The result for the internal mesh point are written as

$$a_m T_{m-1}(t + \Delta t) + b_m T_m(t + \Delta t) + c_m T_{m+1}(t + \Delta t) = s_m(t + \Delta t), \quad (8.11)$$

where

$$a_m = -r_{m-1} \frac{2 k_m(t + \Delta t) \;\; k_{m-1}(t + \Delta t)}{\Delta R_{m-1} \left(k_m(t + \Delta t) + k_{m-1}(t + \Delta t)\right)};$$

$$c_m = -r_m \frac{2 k_m(t + \Delta t) \;\; k_{m+1}(t + \Delta t)}{\Delta R_m \left(k_m(t + \Delta t) + k_{m+1}(t + \Delta t)\right)};$$

$$b_m = -a_m - b_m + \left[\frac{r_m^2 - r_{m-1}^2}{2}\right] \frac{\rho_m(t + \Delta t) \;\; C_m(t + \Delta t)}{\Delta t};$$

$$s_m(t + \Delta t) = \left[\frac{r_m^2 - r_{m-1}^2}{2}\right] \left[\frac{\rho_m(t + \Delta t) \;\; C_m(t + \Delta t)}{\Delta t} \; T_m(t) + q_m(t + \Delta t)\right].$$

The time-discretized equations for the boundary nodes are written in the same way. The nonlinear equations (8.11) are linearized taking the values of thermal conductivity k, density ρ and specific heat C from the previous time step. The resulting three-diagonal system of linear equations is solved using LU decomposition (Ueberhuber, 1995). Setting the term $\frac{1}{\Delta t}$ to zero, the steady-state form of the heat conduction equations is obtained. The same solution procedure is used in this case. To verify the computer code, the steady-state calculations were compared with the analytical solution, which can be derived in the case of constant material properties and flat power distribution (Lahey Jr and Moody,

1977).

8.2 Fluid Dynamics Equations of the Coolant

The equations for the coolant temperature can be written in the following form (Shober et al., 1977):

$$\rho_L(z,t)C_L(z,t)\frac{\partial T_L(z,t)}{\partial t} + G_L(z,t)C_L(z,t)\frac{\partial T_L(z,t)}{\partial z} = \gamma q_L''(z,t) + q_L'''(z,t), \quad (8.12)$$

where

$T_L(z,t)$ is the bulk coolant temperature [°K];

$G_L(z,t)$ is the coolant mass flow rate [kg/m^2 s];

$q_L'''(z,t)$ is the volumetric power generation rate at the coolant [watts/m^3];

$C_L(z,t)$ is the coolant specific heat [J/kg °K];

$\rho_L(z,t)$ is the coolant density [kg/m^3];

γ is the ratio of the surface area of the cladding to the volume of the coolant [m^{-1}];

$q_L''(z,t)$ is the heat flux at the surface of the cladding [watts/m^2].

The equations are discretized at the axial mesh, which coincides with the neutronics mesh. The z-derivative is approximated as

$$\frac{\partial T_L(z,t)}{\partial z} = \frac{2\left[T_{L,i}(t) - T_{L,i}^{in}(t)\right]}{\Delta z_i},$$

where

$T_{L,i}(t)$ is the average coolant temperature of the axial node i;

$T_{L,i}^{in}(t)$ is the inlet coolant temperature of the node i;

Δz_i is the axial height of the node i.

The inlet temperature for the node i is computed using the linear interpolation from the previous node $i-1$ as

$$T^{in}_{L,i}(t) = 2T_{L,i-1}(t) - T^{in}_{L,i-1}(t) \tag{8.13}$$

The resulting equation for the coolant temperature is:

$$\rho_{L,i}(t)C_{L,i}(t)\frac{\partial T_{L,i}(t)}{\partial t} + G_{L,i}(t)C_{L,i}(t)\frac{2}{\Delta z_i}\left[T_{L,i}(t) - T^{in}_{L,i}(t)\right] = \gamma\psi_i(t) + q_{L,i}(t),$$

Using the fully-implicit scheme, the equation (8.13) is written as

$$\begin{aligned} T_{L,i}(t+\Delta t)\left\{\rho_{L,i}(t+\Delta t)C_{L,i}(t+\Delta t)\frac{1}{\Delta t} + G_{L,i}(t+\Delta t)C_{L,i}(t+\Delta t)\frac{2}{\Delta z_i}\right\} = \\ \gamma\psi_i(t+\Delta t) + q_{L,i}(t+\Delta t) + T_{L,i}(t)\left\{\rho_{L,i}(t+\Delta t)C_{L,i}(t+\Delta t)\frac{1}{\Delta t}\right\} + \\ T^{in}_{L,i}(t+\Delta t)G_{L,i}(t)C_{L,i}(t)\frac{2}{\Delta z_i} \end{aligned} \tag{8.14}$$

The nonlinear equation (8.14) is linearized using the values of the coolant density $\rho_{L,i}$ and the coolant specific heat $C_{L,i}$ from the previous time step. The mass flow rate $C_{L,i}$ is constant for all axial nodes. The heat flux at the cladding surface is also approximated using the coolant and cladding temperatures from the previous time step

$$q''_{L,i}(t+\Delta t) \approx h_{L,i}(t)\left[T(R^{out}_{cl}, t) - T_{L,i}(t)\right], \tag{8.15}$$

where the heat transfer coefficient $h_i(t)$ is computed using the inlet coolant temperature $T^{in}_{L,i}(t+\Delta t)$.

To compute the heat transfer coefficient the Dittus-Boelter correlation (Wakil, 1993) is applied:

$$h_{L,i} = 0.023\mathrm{Re}^{0.8}\mathrm{Pr}^{0.4}\frac{k_L}{D_e} \tag{8.16}$$

where

k_L is the thermal conductivity of the coolant [watts/m °K];

Re is the Reynolds number;

Pr is the Prandtl number;

D_e is the equivalent hydraulic diameter.

The Prandtl and Reynolds number are defined as

$$\mathrm{Re} = \frac{D_e G}{\mu}; \quad \mathrm{Pr} = \frac{C_p \mu}{k_L};$$

where

G is the mass flow rate of the coolant [kg/m^2 s];

μ is the viscosity of the coolant [kg/m s].

Final algorithm of the solution of the both heat conduction and fluid dynamics equations is summarized in the Fig. 8.2, where $N\_Channel$ Channel is a number of the coolant channels and NZ is a number of axial layers. Two water packages are available in the SKETCH-N code. The one is adapted from the J-TRAC code (Akimoto et al., 1989), and the other is taken from the JAERI subroutine library JSSL (Inoue et al., 1982). The steady-state calculations are performed in the same way, setting the term $1/\Delta t$ to zero.

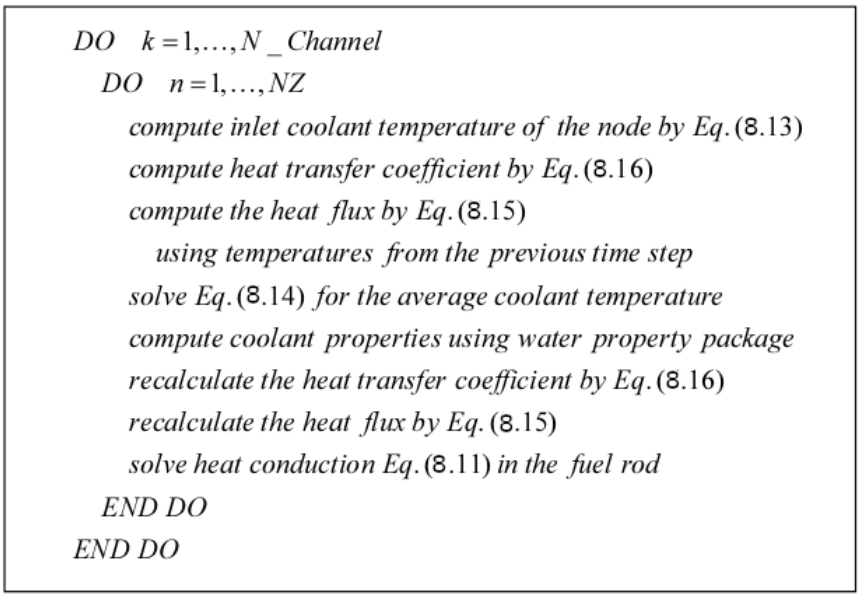


Figure 8.2: Algorithm of the time step calculation by the thermal-hydraulics model

Chapter 9

Coupling interface module

The internal thermal-hydraulics model described in the precious chapter 8 can treat only single-phase coolant flow, thus limiting applications of the code to operational transients in PWR. To use the SKETCH-N code in LWR safety analysis, a coupling with an external thermal-hydraulics code is needed. To simplify the coupling procedure, an interface module has been developed. The module provides data exchange between the coupled codes, data mapping between spatial meshes of the codes and synchronization of time stepping. The module has been applied for a coupling of the SKETCH-N code with the transient analysis codes J-TRAC and TRAC-BF1 (Zimin et al., 1999; Asaka et al., 2000; Zimin et al., 2000). The description of the interface module is given in this chapter.

9.1 Time Stepping and Data Exchange

The coupling and data transfer between the codes is organized using the message-passing library Parallel Virtual Machine (PVM) (Geist, 1994). In the following, we assume that an external thermal-hydraulics model is the transient analysis code TRAC, because a coupling with the J-TRAC code (Akimoto et al., 1989), which is a JAERI version of the TRAC-PF1/MOD1, and with the TRAC-BF1 code (Borkowski et al., 1992) has been already performed (Zimin et al., 1999; Asaka et al., 2000; Zimin et al., 2000). The codes are treated as separate processes and the subroutines based on PVM are responsible for data exchange between the

codes and synchronization of the time stepping. A flow chart of the coupled TRAC/SKETCH-N calculations is illustrated in Fig. 9.1. After performing the input, the TRAC code enrolls into PVM and spawns the child process - SKETCH-N. When SKETCH-N is started, it gets an identification number of the parent process and the codes can communicate to each other sending/receiving messages. At the beginner of a time step, TRAC sends a message to SKETCH-N with thermal-hydraulics reactor data (fuel temperature, coolant density and coolant temperature) and an estimation of the next time step size. SKETCH-N receives the message, selects a new time step size and performs the neutronics calculation. Then, SKETCH-N sends a message to TRAC with the computed power distribution and the used time step size. TRAC receives the message and performs the thermal- hydraulics calculation. The procedure is repeated till the end of the transient.

Actually a time stepping procedure is slightly more complicated as illustrated in Fig 9.2 due to an automatic time step control based on the time step doubling technique. SKETCH performs the calculations using two temporal meshes: the fine temporal mesh, which consists of pairs of the time steps of equal length, and the coarse temporal mesh with a double time step size. A comparison of the power distribution computed on these meshes gives an estimate of the SKETCH time step size. Finally, a time step size of the TRAC/SKETCH code system is selected as a minimum of the time step sizes proposed by the codes. The list of the variables, which transfer between the meshes, is given as follows

from the neutronics mesh into the heat conduction mesh:

heat generation rate of the heat conduction nodes [W];

from the neutronics mesh into the fluid dynamics mesh:

heat generation rate of the fluid dynamics nodes due to the direct coolant heating [W];

from the heat conduction mesh into the neutronics mesh:

average Doppler fuel temperature of the neutronics nodes [°K];

from the fluid dynamics mesh into the neutronics mesh:

average coolant temperature [°K] and coolant density [kg/m] of the neutronics nodes.

Variables transferring from the TRAC code depend on the macro cross section

model of the SKETCH and can be changed for a new reactor type or a new problem.

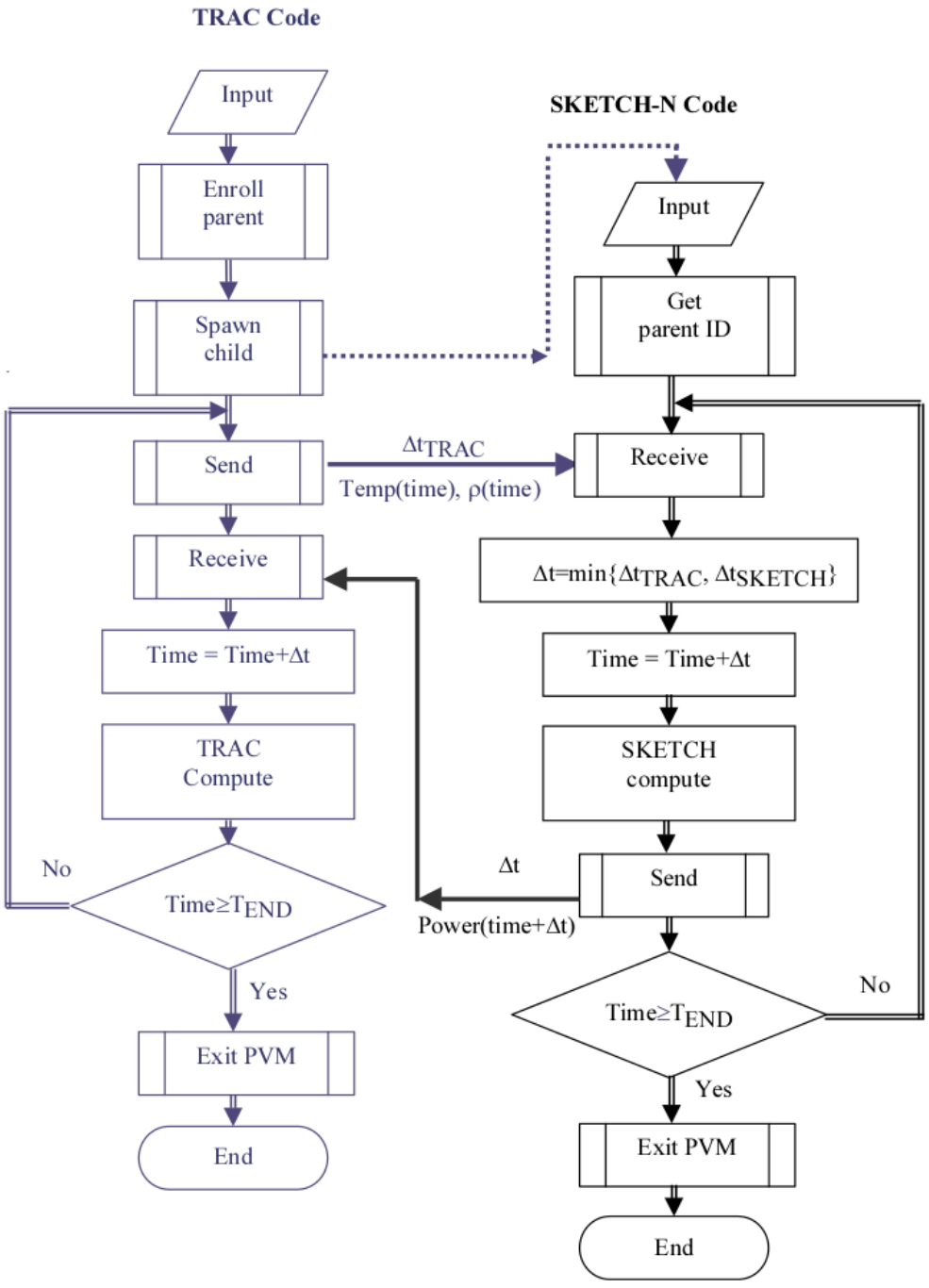


Figure 9.1: Simplified flow chart of the TRAC/SKETCH-N calculations

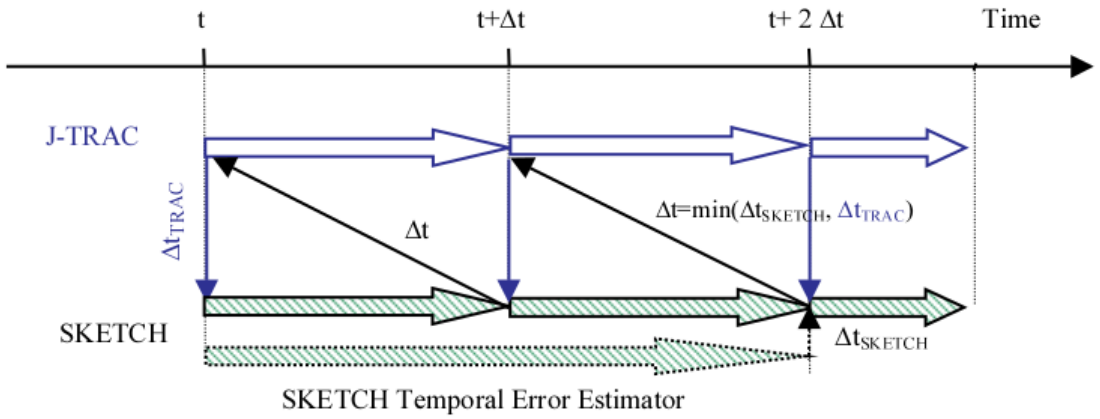


Figure 9.2: Time Stepping of the Coupled TRAC/SKETCH code system

9.2 Mapping of the Data between Spatial Meshes of the Codes

TRAC and SKETCH-N codes usually perform calculations using different spatial meshes. Additionally TRAC code applies axially staggered meshes for the fluid dynamics and heat conduction models. An example of the axial meshes used in the codes is given in Fig. 9.3. The coupled TRAC/SKETCH-N calculations require a mapping of the data between the neutronics, heat conduction and fluid dynamics meshes.

The mapping procedure between the meshes is based on the mapping matrix approach developed for the PARCS code (Downar et al., 1997a). Let us consider the mapping of the heat generation rate computed by the SKETCH code on the neutronics spatial mesh into the heat conduction mesh and into the fluid dynamics mesh. The mapping is performed using the two mapping matrices: S^{HC}_{NEUT} and S^{FD}_{NEUT}. Even the spatial meshes of SKETCH-N and TRAC codes can be very different, they share a one common feature: the 3D spatial mesh of the both codes is a tensor product of the 2D spatial mesh in radial plane and the 1D axial mesh. As a result the 3D mapping matrix can be also expressed as a tensor product of the 2D and 1D mapping matrices:

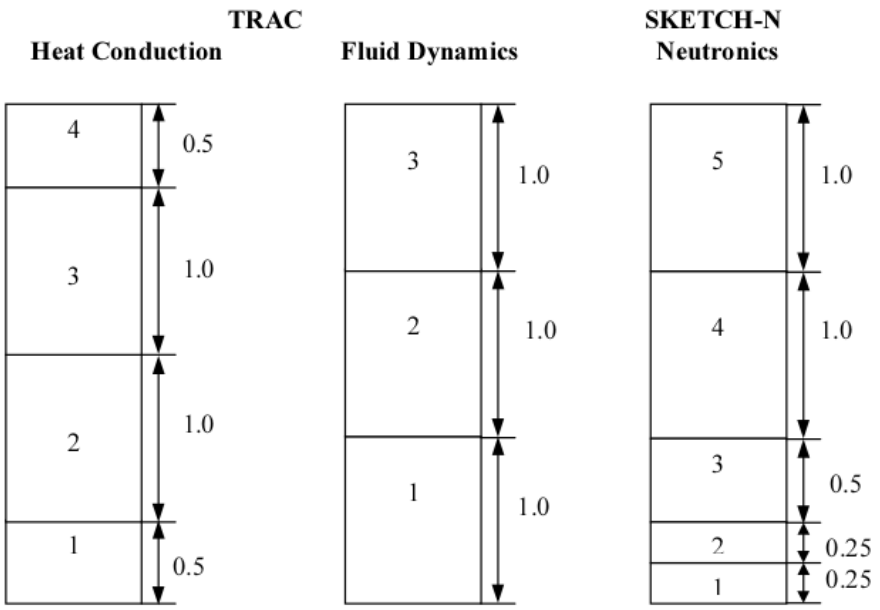


Figure 9.3: An example of the axial meshes in the TRAC and SKETCH-N codes

$$S = S2D \otimes S1D \tag{9.1}$$

where

$\otimes$ is a direct (Kronecker) matrix product;

$S2D$ is the mapping matrix describing a correspondence of 2D radial meshes;

$S1D$ is the mapping matrix for axial meshes.

These mapping matrices $S2D$ and $S1D$ are defined by a code user for two pairs of the spatial meshes: neutronics mesh $\Rightarrow$ heat conduction mesh and neutronics mesh $\Rightarrow$ fluid dynamics mesh. Let us consider the calculation of the mapping matrix for two 1D axial meshes: the neutronics mesh z with J nodes, j-th node is the interval $[z_j, z_{j+1}]$ and the heat conduction mesh Z with I nodes, i-th node is the interval $[Z_i, Z_{i+1}]$. The mapping matrix $S1D$ for these two meshes is computed as

$$S1D_{ij} = \frac{|[z_j, z_{j+1}] \cap [Z_i, Z_{i+1}]|}{|[z_j, z_{j+1}]|} \tag{9.2}$$

where

$[z_j, z_{j+1}] \cap [Z_i, Z_{i+1}]$ is an intersection of the intervals $[z_j, z_{j+1}]$ and $[Z_i, Z_{i+1}]$;

$|\circ|$ is the function equals to the length of the interval.

For example, the mapping matrices for the axial meshes defined in Fig. 9.3 are given as

$$\mathrm{STD}^{RC}_{NEUT} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0.5 \end{pmatrix}; \quad \mathrm{STD}^{FD}_{NEUT} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

where

$S1D^{HC}_{NEUT}$ is the mapping matrix from the neutronics mesh into the heat conduction mesh;

$S1D^{FD}_{NEUT}$ is the mapping matrix from the neutronics mesh into the fluid dynamics mesh.

The calculation of the mapping matrices for 2D spatial meshes is very similar, the only difference that the area of the nodes is taken instead of the length of the nodes. Because the mapping matrices are sparse, the compressed row storage (CRS) format is used for their input and storage (Barrett et al., 1994). According to the definition, see Eq. (9.2), the mapping matrices satisfy the following condition

$$\sum_{i=1}^{I} S_{ij} = 1$$

This condition providing the conservation law is checked in the code when an input of the mapping matrices is performed.

Using the defined mapping matrices the heat generation rate on the heat conduction mesh $P\,[HC]$ is computed as

$$P\,[HC] = (1-\alpha) S^{HC}_{NEUT} P\,[NEUT]\,,$$

where

α is a part of the energy directly deposited into the coolant;

$P\,[NEUT]$ is the heat generation rate computed by the SKETCH-N code on the neutronics spatial mesh;

S^{HC}_{NEUT} is the 3D mapping matrix from the neutronics mesh into the heat conduction mesh computed using Eq. (9.1).

The heat generation rate on the fluid dynamics mesh $P\,[FD]$ is computed as

$$P\,[FD] = \alpha S^{FD}_{NEUT} P\,[NEUT]\,,$$

where S^{FD}_{NEUT} is the 3D mapping matrix from the neutronics mesh into the fluid dynamics mesh computed using Eq. (9.1).

The data mapping from the TRAC spatial meshes into the neutronics mesh is performed in a similar fashion. The Doppler fuel temperature computed by the TRAC code on the heat conduction mesh $TF\,[HC]$ is mapped into the neutronics mesh as

$$TF\,[NEUT] = S^{NEUT}_{HC} TF\,[HC]\,,$$

where

$TF\,[NEUT]$ is the Doppler fuel temperature on the neutronics mesh;

S^{NEUT}_{HC} is the mapping matrix from the heat conduction mesh into the neutronics mesh.

If geometry is specified consistently for the both codes, the mapping matrix S^{NEUT}_{HC} is simply a transpose of the mapping matrix S^{HC}_{NEUT} :

$$S^{NEUT}_{HC} = \left(S^{HC}_{NEUT}\right)^T$$

The treatment of the coolant temperature and coolant density is slightly more involved. Due to the donor-cell method used in the TRAC code, the computed coolant temperatures and coolant densities are the values at the interfaces of the fluid dynamics nodes. To get the node-average values of the coolant temperature on the fluid dynamics axial mesh the linear interpolation is applied as follows

$$TC^n\left[FD\right] = 0.5\left(TC^n\left[FD\_TRAC\right] + TC^{nn}\left[FD\_TRAC\right]\right), n = 1, \ldots, N;$$

where

n is an axial index of the node;

$TC\left[FD\right]$ is the node-average value of the coolant temperature;

$TC\left[FD\_TRAC\right]$ is the coolant temperature computed by the TRAC code;

$nn = \begin{cases} n-1, & \text{if } V_z \geq 0 \\ n+1, & \text{if } V_z < 0 \end{cases}$, where V_z is the coolant velocity in axial direction.

The coolant density is interpolated in the same way. Then, the node-averaged values of the coolant density and the coolant temperature are mapped into the neutronics mesh in the similar way as described above for the fuel temperature.

Chapter 10

Conclusions

10.1 Summary of the SKETCH-N Code Features

The basic features of the nodal three-dimensional neutron kinetics code SKETCH-N are summarized in the following:

- diffusion approximation;
- 3D, 2D and 1D reactor models in Cartesian and hexagonal-z geometries;
- arbitrary number of neutron energy groups and delayed neutron precursors;
- transverse-integrated polynomial, semi-analytic and analytic nodal methods with quadratic leakage approximation for spatial discretization;
- nonlinear iteration procedure for a solution of the nodal equations;
- fully-implicit scheme with analytic integration of the delayed neutron precursors for time discretization;
- adaptive time step control based on the step doubling technique;
- inverse iterations with Wieland shift accelerated by Chebyshev polynomials for the steady-state eigenvalue problems;
- adaptive Chebyshev acceleration procedure for the neutron kinetics problems;

- block symmetric Gauss-Seidel method as a preconditioner;
- consistent point kinetics option, where the point kinetics parameters are computed at each time step using the values of macro cross sections;
- internal thermal-hydraulics model for PWR and VVER operational transients with single-phase coolant flow;
- interface module based on the message passing library PVM for the coupling with external thermal-hydraulics codes, such as TRAC.

10.2 Outline of the Code Verification

This report describes only the SKETCH-N models, verification results of the LWR benchmark problem calculations are given in the papers cited below. The computed steady-state LWR neutronics problems include:

- classical 2D & 3D International Atomic Energy Agency (IAEA) PWR problems;
- 2D Biblis PWR checker-board-loaded core;
- 2D Zion-1 PWR problem with explicit modeling of the baffle;
- 2D 4-group Koeberg PWR checker-board-loaded core with realistic cross sections including up-scattering;
- 2D IAEA benchmark problems modified for hexagonal geometry;
- 2D and 3D two-group problems for VVER-440 and VVER-1000 reactors;
- 2D four-group problem for a VVER-1000 reactor;
- 2D two-group problem for a heavy-water-moderated reactor in hexagonal geometry.

The calculation results presented in the works (Zimin, 1997; Zimin et al., 1998; Zimin and Baturin, 2002) demonstrated the program's high accuracy. The errors

in the multiplication factor for all calculated problems were less than 0.025%, and the errors in assembly power did not exceed 2.5%.

The neutron kinetics module has been verified against 3D Langenbuch-Maurer-Werner (LMW) operational transient in a small PWR model, 2D & 3D super-prompt-critical rod drop accident in BWR (Zimin and Ninokata, 1998) and the AER2 benchmark problem on peripheral control rod ejection in a VVER-440 reactor (Zimin and Baturin, 2001). The SKETCH-N code has been coupled with the J-TRAC (TRAC-PF1) and TRAC-BF1 codes. The J-TRAC/SKETCH-N code system has been verified against NEACRP PWR rod ejection benchmark and NEA/NSC PWR benchmark on uncontrolled rod withdrawal at zero power (Zimin et al., 1999; Asaka et al., 2000). Assessment of the TRAC-BF1/SKETCH-N code system has been performed by NEACRP BWR cold water injection benchmark (Asaka et al., 2000; Zimin et al., 2000).

To outline the SKETCH-N accuracy we give some results of the steady-state LWR neutronics calculations in Table 10.1 taken from (Zimin et al., 1998).

Table 10.1: SKETCH-N numerical results of the steady-state LWR benchmark problems.

Problem, Mesh	Nodal Method	E_{max}/E_{av}, %	Position of E_{max} (X,Y)	k_{eff}	Reference k_{eff}	Δk_{eff} pcm	No. of Iterations: Nonlinear/ Outers
IAEA-2D 9x9	NEM[1]	-2.05/-	(6,6)	1.0295	1.0296	-	-/32
	PNM	-1.8/0.7	(6,6)	1.02950		-8	
	SANM	0.5/0.2	(6,5)	1.02956	1.029585[2]	-3	9/28
	ANM	1.1/0.3	(5,7)	1.02962		+3	
BIBLIS-2D 9x9	NEM[3]	1.25/0.49	-	1.02528	1.02512	-	-
	PNM	-1.2/0.4	(1,8)	1.02521		+10	10/30
	SANM	1.8/0.5	(6,6)	1.02526	1.025110[3]	+15	
	ANM	2.2/0.7	(6,6)	1.02532		+21	
BWR-2D 11x11	NEM[4]	1.36/0.26	(1,1)	0.996329		-	-/22
	PNM	1.9/0.6	(1,1)	0.99626	0.99636[4]	-10	8/28
	SANM	0.5/0.2	(1,1)	0.99635		-1	
	ANM	0.2/0.1	(9,6)	0.99641		+5	
IAEA-3D 9x9x19	NEM[1]	-1.62/-	(6,6)	1.0290	1.0290[1]	-	-/24
	PNM	-1.8/0.7	(6,6)	1.02899		-8	
	SANM	0.4/0.2	(7,5)	1.02905	1.029074[5]	-2	8/38
	ANM	1.1/0.3	(7,5)	1.02912		+4	
BWR-3D 12x12x14	NEM[4]	1.17/0.22	(1,1)	0.996361		-	7/21
	PNM	1.4/0.4	(1,1)	0.99633	0.996368[4]	-4	7/37
	SANM	0.4/0.1	(1,1)	0.99637		0	
	ANM	0.5/0.1	(8,8)	0.99643		+6	
LMW 6x6x10	NEM[1]	-1.2/-	(5,4)	0.99958		-	-/19
	PNM	-1.3/0.4	(5,4)	0.99958	0.99966[1]	-8	7/26
	SANM	0.4/0.1	(4,4)	0.99971		+5	
	ANM	0.5/0.2	(4,4)	0.99977		+11	

[1] the results are taken from (Bandini, 1990);

[2] the reference results are taken from (Müller and Weiss, 1991);

[3] the NEM results are taken from (Greenman et al., 1979);

[4] the results are taken from (Gehin, 1992);

[5] the reference results computed by the SKETCH-N code with SANM on spatial

mesh 34x34x38 (5 cm in X-Y directions and 10 cm in Z direction).

The presented problems include

- 2D & 3D IAEA PWR problems (Argonne National Laboratory, 1977),
- 2D BIBLIS PWR checker-board-loaded core (Müller and Weiss, 1991); and initial conditions of the neutron kinetics benchmarks:
- 3D LMW LWR problem (Langenbuch et al., 1977);
- 2D and 3D BWR LRA problems (Argonne National Laboratory, 1977).

The SKETCH-N results are given using the polynomial (PNM), semi-analytic (SANM) and analytic (ANM) nodal methods. The results of the nodal expansion method (NEM) taken from the literature are added for a comparison. All calculations are performed using one node per fuel assembly in x-y plane. The reference solutions for these problems are computed using a fine spatial mesh or high-order nodal methods. A comparison is performed for the multiplication factor k and the assembly-averaged 2D power densities. The errors in k are given in pcm (1 pcm=0.001%), the maximum E_{max} and average E_{av} errors in 2D power distribution are given in percents. The results show that all the nodal methods provide acceptable accuracy for these problems. The maximum errors of the assembly-averaged power density are less than 2.5% and the errors in eigenvalue are about 10-15 pcm. The accuracy of the semi-analytic and analytic methods is usually better than that of PNM, except for the BIBLIS-2D checker-board-loaded core. Table 10.1 also shows a number of nonlinear iterations and a total number of outer iterations. The iterative solution techniques are efficient, the number of the nonlinear iterations does not exceed 10 for all the given problems, and the total number of the outer iterations does not exceed 50.

Some results of test problem calculations for hexagonal geometry are presented in Table 10.2 (Zimin and Baturin, 2002).

Table 10.2: Calculation errors of assembly power (δP^{2D}_{max}, δP^{2D}_{av}) and multiplication factor (δk_{eff}) using SKETCH-N code for 2D and 3D test problems in hexagonal geometry.

Problem	δP^{2D}_{max}, (%)	δP^{2D}_{av}, (%)	δk_{eff}, (pcm=10^{-5})
2D VVER-440	0.5	0.2	24
2D VVER-1000 (albedo 0.5)	0.8	0.3	7
2D VVER-1000 (albedo 0.125)	2.1	0.6	0
2D VVER-1000 (4 groups)	0.5	0.2	-2
2D IAEA (no reflector, albedo 0.5)	0.6	0.2	4
2D IAEA (no reflector, albedo 0.125)	1.3	0.5	-6
HWR	0.5	0.1	8
3D VVER-440	0.7	0.3	25
3D VVER-1000	0.8	0.4	26

A comparison of the reactor power calculated by the SKETCH-N code with results from other VVER codes for the AER2 benchmark problem of control rod ejection from a cold state of the VVER-440 reactor is presented in Fig. 10.1 (Zimin and Baturin, 2001).

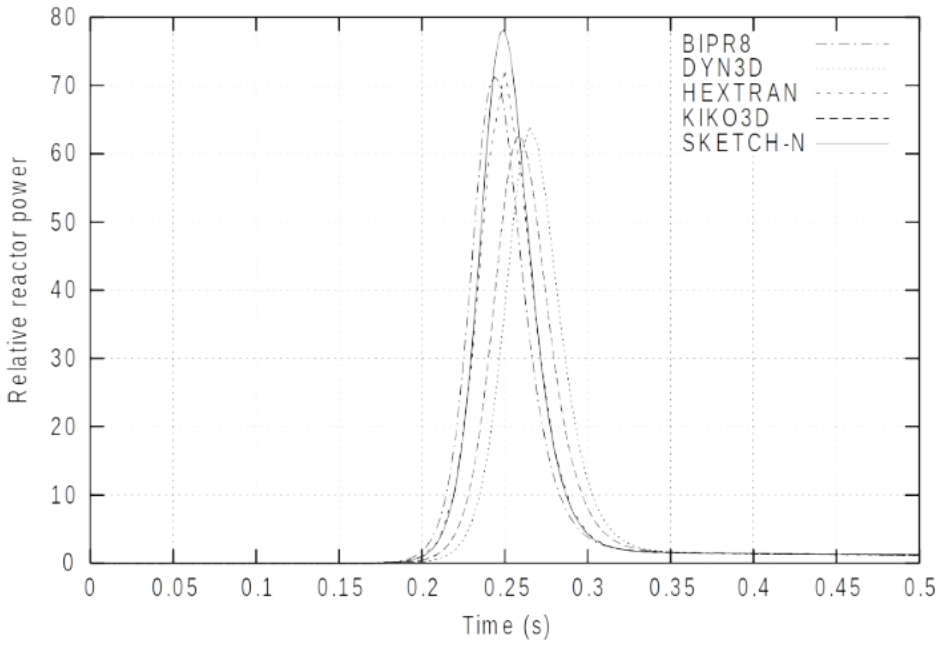


Figure 10.1: Reactor power over time for the AER2 benchmark problem

Accuracy of the neutron kinetics module is illustrated by the NEACRP PWR rod ejection benchmark (Finnemann and Galati, 1992). All six cases of the benchmark are computed by the J-TRAC/SKETCH-N code system (Zimin et al., 1999). The cases A1, B1 and C1 are defined at the hot zero power (HZP) initial conditions, while the cases A2, B2, and C2 are specified at the full power (FP). Tables 10.1–10.6 show respectively the steady-state and transient results taken from (Zimin et al., 1999).

Table 10.3: J-TRAC/SKETCH-N steady-state results of the PWR NEACRP rod ejection benchmark for Hot Zero Power (HZP) conditions (upper values), PANTHER reference solution (middle values) and deviations (lower values).

Parameter	**Case A1**	**Case B1**	**Case C1**
Critical Boron Concentration, ppm	561.9 561.20 +0.7 ppm	1248.7 1247.98 +0.7 ppm	1129.2 1128.29 +0.9 ppm
3D Nodal Power Peaking Factor	2.8770 2.8792 -0.08 %	1.9333 1.9330 +0.02 %	2.1873 2.1867 +0.02 %
Control Rod Worth, Pcm	822.2 824.31 -2 pcm	826.4 826.18 +0.2 pcm	948.4 949.09 -0.7 pcm

Table 10.4: J-TRAC/SKETCH-N steady-state results of the PWR NEACRP rod ejection benchmark for Full Power (FP) conditions (upper values), PANTHER reference solution (middle values) and deviations (lower values).

Parameter	**Case A2**	**Case B2**	**Case C2**
Critical Boron Concentration, ppm	1155.4 1156.63 -1.2 ppm	1182.2 1183.83 -1.6 ppm	1155.6 1156.63 -1 ppm
3D Nodal Power Peaking Factor	2.2223 2.2073 +0.7 %	2.1107 2.0954 +0.7 %	2.2213 2.2073 +0.6 %
Control Rod Worth, Pcm	85 91.58 -7 pcm	92 99.45 -7 pcm	74 79.23 -5 pcm

Table 10.5: J-TRAC/SKETCH-N transient results of the PWR NEACRP rod ejection benchmark for Hot Zero Power (HZP) conditions (upper values), PANTHER reference solution (middle values) and deviations (lower values).

Parameter	**Case A1**	**Case B1**	**Case C1**
Time to the Power Peak, s	0.547 0.5375 +0.01 s	0.525 0.5225 +0.003 s	0.2732 0.2712 +0.002 s
Power at the Peak, %	123.0 126.78 -3.0 %	229.7 231.51 -0.8 %	437.0 441.12 -1 %
Power at 5 s, %	19.0 19.69 -3.5 %	31.0 31.97 -3.0 %	14.3 14.60 -2 %
Average Doppler Temperature at 5 s, °C	324.1 324.89 -0.8 °C	349.2 349.96 -0.8 °C	315.8 315.91 -0.1 °C
Max Fuel Centerline Temperature at 5 s, °C	672 679.30 -7 °C	556 559.66 -4 °C	675 674.20 +0.8 °C
Coolant Outlet Temperature at 5 s, °C	293.1 293.22 -0.1 °C	297.6 297.72 -0.1 °C	291.54 291.53 0.01 °C

Table 10.6: J-TRAC/SKETCH-N transient results of the PWR NEACRP rod ejection benchmark for Full Power (FP) conditions (upper values), PANTHER reference solution (middle values) and deviations (lower values).

Parameter	**Case A2**	**Case B2**	**Case C2**
Time to the Power Peak, s	0.095 0.095 -	0.095 0.1 -0.005 s	0.095 0.095 -
Power at the Peak, %	108.0 108.3 -0.3 %	106.4 106.4 -	107.25 107.34 +0.1 %
Power at 5 s, %	103.3 103.62 -0.3 %	103.8 103.94 -0.1 %	102.9 103.14 -0.2 %
Average Doppler Temperature at 5 s, °C	554.8 555.16 -0.4 °C	552.8 552.39 +0.4 °C	553.7 553.90 -0.2 °C
Max Fuel Centerline Temperature at 5 s, °C	1688 1679.6 +8 °C	1556 1576.10 -20 °C	1724 1723.80 +0.2 °C
Coolant Outlet Temperature at 5 s, °C	325.7 324.90 +0.8 °C	325.9 324.98 +0.9 °C	325.6 324.77 +0.8 °C

The J-TRAC/SKETCH-N calculations are performed using the semi-analytic nodal method with 4 nodes per fuel assembly in x-y plane. Automatic time step control is used in the transient calculations. The results are compared with the reference solution computed by the PANTHER code using fine spatial and temporal meshes (Knight and Bryce, 1997). Both the steady-state and transient results show excellent agreement with the reference PANTHER results. Please, note that similar results can be also computed by the stand-alone SKETCH-N code with the internal thermal-hydraulics model. The presented results and the other results given in the cited papers demonstrate acceptable accuracy and efficiency of the SKETCH-N code for LWR safety analysis and design.

Bibliography

Akimoto, H. et al. (1989). 18-th report on the best-estimate code development. J-TRAC user manual (version 62.1). Technical Report JAERI-Memo-63-478, JAERI. [In Japanese].

Al-Chalabi, R., Turinsky, P., Faure, F.-X., Sarsour, H., and Engrand, P. (1993). NESTLE: a nodal kinetics code. *Transactions of the American Nuclear Society;(United States)*, 68(CONF-930601-).

Argonne National Laboratory (1977). *Argonne code center benchmark problem book*. ANL-7416, Suppl. 2.

Asaka, H., Zimin, V. G., Iguchi, T., and Anoda, Y. (2000). Coupling of the thermal hydraulics TRAC codes with 3D neutron kinetics code SKETCH-N. In *Preliminary Proc. of the OECD/CSNI Workshop on Advanced Thermal-Hydraulic and Neutronics Codes: Current and Future Applications, Barcelona, Spain, 10-13 April, 2000*, pages 1–15.

Aviles, B. (1993). Development of a variable time-step transient NEW code: SPANDEX. *Transactions of the American Nuclear Society;(United States)*, 68(CONF-930601-).

Bandini, B. R. (1990). *A three-dimensional transient neutronics routine for the TRAC-PF1 reactor thermal hydraulic computer code*. PhD thesis, Pennsylvania State University, LA-SUB-95-21.

Barrett, R., Berry, M., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and Van der Vorst, H. (1994). *Templates for the solution of linear systems: building blocks for iterative methods*. SIAM.

Borkowski, J., Smith, K., Hagrman, D., Kropaczek, D., Rhodes III, J., and Esser, P. (1996). SIMULATE-3K simulation of the ringhals 1 BWR stability measurements. In *Proc. Int. Conf. on the Physics of Reactors (PHYSOR 96), Mito, Ibaraki, Japan, 16-20 September, 1996, p. J-121.*

Borkowski, J. A. et al. (1992). *TRAC-BF1/MOD1: An advanced best-estimate computer program for BWR accident analysis. Model description. NUREG/CR-4356, EGG-2626, Vol. 1.*

Cameron, I. R. (1982). *Nuclear fission reactors.* Plenum Press, New York.

Crouzet, N. and Turinsky, P. J. (1996). A second-derivative-based adaptive time-step method for spatial kinetics calculations. *Nuclear science and engineering*, 123(2):206–214.

Downar, T., Barber, D., Ebert, D., and Mousseau, V. (1997a). Three-dimensional spatial kinetics for coupled thermal-hydraulics systems analysis codes. presentation on 25 water reactor safety meeting, 22 october 1997.

Downar, T., Joo, H. G., and Jiang, G. (1997b). A hybrid ANM/NEM interface current technique for the nonlinear nodal calculation. In *Proc. Joint Int. Conf. Math. Comp*, pages 124–133.

Engrand, P., Maldonado, G., Al-Chalabi, R., and Turinsky, P. (1994). Non-linear iterative strategy for NEM refinement and extension. In *Transactions of American Nuclear Society 65, 221.*

Ferguson, D. and Derstine, K. (1977). Optimized iteration strategies and data management considerations for fast reactor finite difference diffusion theory codes. *Nuclear Science and Engineering*, 64(2):593–604.

Finnemann, H. (1975). A consistent nodal method for the analysis of space-time effects in large LWR's. In *Proc. Joint NECRP/CSNI Spesialists' Mtg. New Developments in Three-Dimensional Neutron Kinetics and Review of Kinetics Benchmark Calculations, Garching, FRG, January 22-24, 1975*, page 13.

Finnemann, H., Bennewitz, F., and Wagner, M. (1977). Interface current techniques for multidimensional reactor calculations. *Atomkernenergie;*, 30(2):123.

Finnemann, H. and Galati, A. (1992). NEACRP 3-D LWR core transient benchmark: final specifications. Technical Report NEACRP-L-335 (Revision 1), January 1992.

Gantmacher, F. R. (1959). *Theory of matrices*, volume I, chapter V. Chelsea Publishing Company, New York.

Gehin, J. C. (1992). *An analytic nodal method for solving the two-group, multidimensional, static and transient neutron diffusion equations.* PhD thesis, Nuclear Engineering, Massachusetts Institute of Technology, DOE/OR/0033-T557.

Geist, A. (1994). *PVM 3 user's guide and reference manual, ORNL/TM-12187, September 1994.*

Golub, G. and Van Loan, C. (1996). *Matrix computations*, chapter 11. John Hopkins University Press, Baltimore and London, 3rd edition.

Greenman, G., Smith, K., and Henry, A. F. (1979). Recent advances in an analytic nodal method for static and transient reactor analysis. In *Proc. of Topl. Meet. On Computational Methods in Nuclear Engineering, Williamsburg, Virginia, April 23-25, 1979.* American Nuclear Society.

Hageman, L. A. and Young, D. M. (1981). *Applied iterative methods.* Applied Iterative Methods, Academic Press, New York.

Hairer, E. S., P., N., and G., W. (1987). *Solving ordinary differential equations I: nonstiff problems.* Springer-Verlag, Berlin.

Henry, A. F. (1975). *Nuclear reactor analysis.* MIT Press, Cambridge, Massachusetts and London.

Inoue, S., Fujimura, T., Tsutsui, T., and Nishida, T. (1982). *JSSL-JAERI scientific subroutine library.* 3rd edition.

Johansson, M. (1994). Stability benchmark. questions and news, Note 94/L/1051, september 9, 1994.

Joo, H.-G., Downar, T., and Barber, D. (1996). Methods and performance of a parallel reactor kinetics code PARCS. In *Proc. Int. Conf. on the Physics of Reactors PHYSOR 96, Mito, Ibaraki, Japan, 16-20 September, 1996, p. J-42.*

Knight, M. P. and Bryce, P. (1997). Derivation of a refined PANTHER solution to the NEACRP PWR rod ejection transients. In *Proc. of the Joint Int. Conf. on Mathematical Methods and Supercomputing for Nuclear Applications, Saratoga Springs, N.Y., October 5-9, 1997*, volume 1, pages 302–313. American Nuclear Society.

Lahey Jr, R. and Moody, F. (1977). The thermal-hydraulics of a boiling water nuclear reactor. *American Nuclear Society.*

Langenbuch, S., Maurer, W., and Werner, W. (1977). Coarse-mesh flux-expansion method for the analysis of space-time effects in large light water reactor cores. *Nuclear Science and Engineering*, 63(4):437–456.

Lefvert, T. (1994). BWR stability benchmark final specifications. *NEA/NSC/DOC (94) 15.*

Marchuk, G. and Lebedev, V. (1981). *Numerical methods in neutron transport theory.* Atomizdat, Moscow. [In Russian].

Moler, C. and Van Loan, C. (1978). Nineteen dubious ways to compute the exponential of a matrix. *SIAM review*, 20(4):801–836.

Müller, E. and Weiss, Z. (1991). Benchmarking with the multigroup diffusion high-order response matrix method. *Annals of Nuclear Energy*, 18(9):535–544.

Nakamura, S. (1977). *Computational methods in engineering and science, with applications to fluid dynamics and nuclear systems.* John Wiley and Sons, Inc., New York.

Noh, J. M., Pogosbekyan, L., Kim, Y. J., and Joo, H. K. (1997). A general approach to multigroup extension of the analytic function expansion nodal

method based on matrix function theory. In *Proc. 1996 Joint Intl. Conf. Mathematical Methods and Super Computing for Nuclear Applications, Saratoga Springs, New York, Octorber*, pages 6–10.

Patankar, S. (1980). *Numerical heat transfer and fluid flow.* Hemisphere Publishing Corporation.

Pease, M. C. and Iii, C. (1965). *Methods of matrix algebra*, chapter VI. Academic Press, New York.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in Fortran 77*, volume 1. Cambridge University Press, Cambridge, 2 edition.

Shikhov, S. B. and Troyanskii, V. B. (1983). *Nuclear reactor theory*, volume 2, Transport Theory. Energoatomizdat, Moscow. [In Russian].

Shober, R., Sims, R., and Henry, A. (1977). Two nodal methods for solving time-dependent group diffusion equations. *Nuclear Science and Engineering*, 64(2):582–592.

Smith, K. S. (1979). An analytic nodal method for solving the two-group, multidimensional, static and transient neutron diffusion equations. Master's thesis, Massachusetts Institute of Technology.

Stacey, W. M. (1969). *Space-time nuclear reactor kinetics.* Academic Press, New York.

Turinsky, P. J., Al-Chalabi, R. M., Engrand, P., Sarsour, H. N., Faure, F. X., and Guo, W. (1994). NESTLE: Few-group neutron diffusion equation solver utilizing the nodal expansion method for eigenvalue, adjoint, fixed-source steady-state and transient problems. Technical Report EGG-NRE-11406, EG and G Idaho, Inc., Idaho Falls, ID (United States); Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Ueberhuber, C. W. (1995). *Numerical computation: methods, software and analysis, Vol. I and II.* Springer-Verlag, Berlin, New York.

Varga, R. S. (1962). *Matrix iterative analysis.* Prentice-Hall, Englewood Cliffs, New Jersey.

Wachspress, E. L. (1966). *Iterative solution of elliptic systems: And applications to the neutron diffusion equations of reactor physics.* Prentice-Hall, Englewood Cliffs, New Jersey.

Wakil, M. M. (1993). *Nuclear heat transport.* American Nuclear Society.

Wolfram, S. (1996). *The mathematica book.* Cambridge University Press, Cambridge, 3rd edition.

Zimin, V. G. (1997). *Nodal neutron kinetics models based on nonlinear iteration procedure for LWR analysis.* PhD thesis, Research Laboratory for Nuclear Reactors, Tokyo Institute of Technology, August 1997.

Zimin, V. G., Asaka, H., Anoda, Y., and Enomoto, M. (1999). Verification of the J-TRAC code with 3D neutron kinetics model SKETCH-N for PWR rod ejection analysis. In *Proc. of the 9 International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-9), San Francisco, California, October 3-8, 1998.* American Nuclear Society. CD-ROM.

Zimin, V. G., Asaka, H., Anoda, Y., Kaloinen, E., and Kyrki-Rajämaki, R. (2000). Analysis of NEACRP 3D BWR core transient benchmark. In *Proc. of the 4th Intl. Conf. on Supercomputing in Nuclear Application SNA-2000, Tokyo, Japan, September 4-7, 2000.* Japan Atomic Energy Research Institute. CD-ROM.

Zimin, V. G. and Baturin, D. M. (2001). Analysis of the VVER-440 AER2 rod ejection benchmark by the SKETCH-N code. In *Proc. of 11th Symposium of AER on VVER Reactor Physics and Reactor Safety, Csopak, Hungary, September 24 - 28, 2001.* CD-ROM.

Zimin, V. G. and Baturin, D. M. (2002). Polynomial nodal method for solving neutron diffusion equations in hexagonal-z geometry. *Annals of Nuclear Energy*, 29(9):1105–1117.

Zimin, V. G. and Ninokata, H. (1998). Nodal neutron kinetics model based on nonlinear iteration procedure for LWR analysis. *Annals of Nuclear Energy*, 25:507–528.

Zimin, V. G., Ninokata, H., and Pogosbekyan, L. R. (1998). Polynomial and semi-analytic nodal methods for nonlinear iteration procedure. In *Proc. of the Int. Conf. on the Physics of Nuclear Science and Technology (PHYSOR98), Long Island, New York, October 5-8, 1998*, volume 2, pages 994–1002. American Nuclear Society.

Appendix A

Conformal Mapping of a Hexagon to a Rectangle

To solve the neutron diffusion equation defined on the resulting rectangle, we don't need to know the explicit form of the function $\mathbb{W} = f(\mathbb{Z})$, it's sufficient to find the distortion function $g(u, v) = \mid \frac{\partial \mathbb{Z}}{\partial \mathbb{W}} \mid$.

The conformal mapping of a hexagon to a rectangle can be obtained through successive conformal mappings.

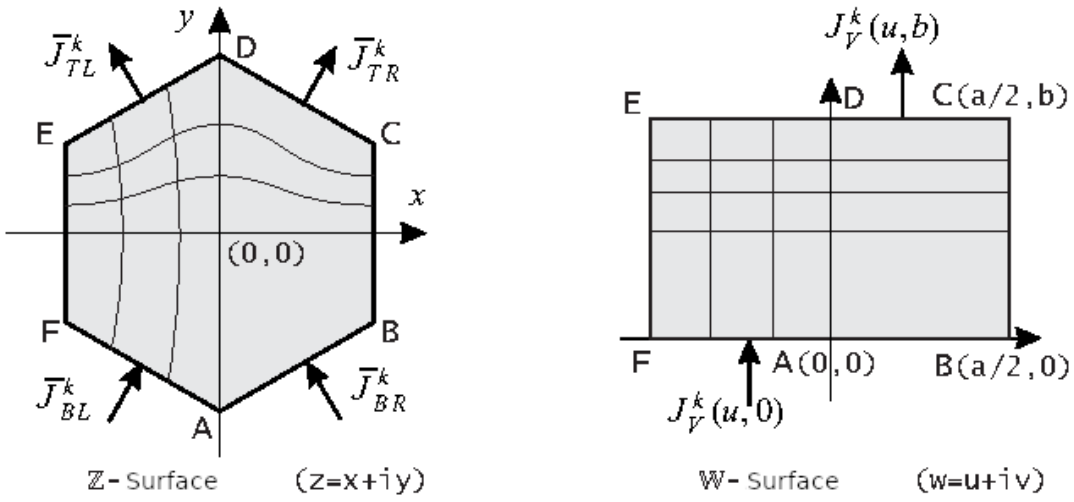


Figure A.1: Problem domain on $\mathbb{Z}$ and $\mathbb{W}$ planes

The function mapping a unit circle defined on the complex plane $\mathbb{T}$ to a hexagon with side length R (Fig.A.1) can be written as:

$$z = \frac{R}{R_0}\int_0^t 1/[(1+p^6)]^{1/3}\mathrm{d}p, \tag{A.1}$$

where R_0 is a normalization factor [1]:

$$R_0 = \frac{2}{\sqrt{3}}\int_0^1 1/[(1+p^6)]^{1/3}\mathrm{d}p \cong 1.112913.$$

The function mapping the upper half-plane $\mathbb{S}$ to the unit circle can be written as:

$$t = i[(s - i\sqrt{3})/(s + i\sqrt{3})] \tag{A.2}$$

Under these transformations, the vertices of the hexagon map to the points $\{0, 1, 3, \infty, -3, -1\}$.

To map the upper half-plane $\mathbb{S}$ to a rectangle whose vertices correspond to the

[1] All numerical values of special functions and integrals were obtained using the *Mathematica* software package.

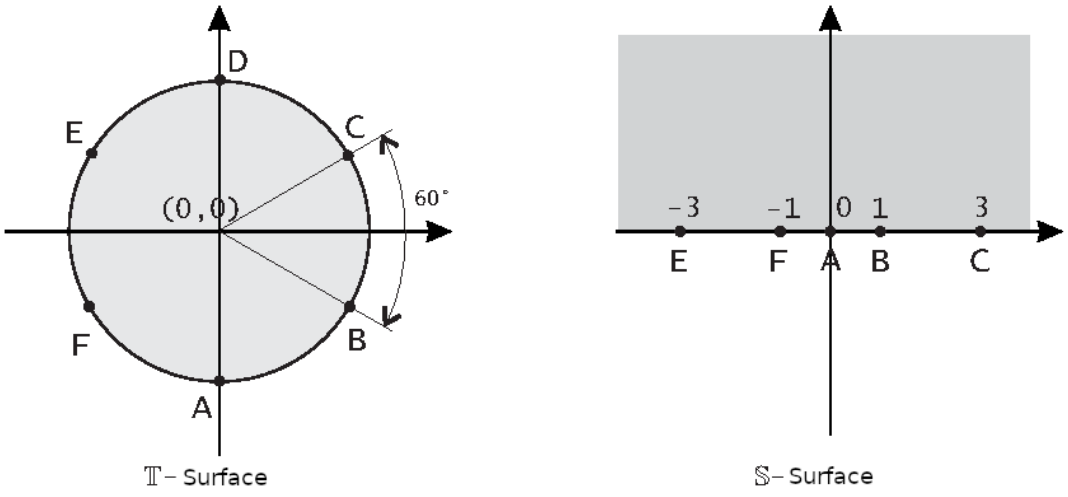


Figure A.2: Problem domain on $\mathbb{T}$ and $\mathbb{S}$ planes

hexagon's vertices, we can use the Schwarz-Christoffel formula:

$$w = \int_0^s [(1-p^2)^{1/2}(1-p^2/9)^{1/2}]^{-1}\mathrm{d}p \tag{A.3}$$

Differentiating equations (A.1, A.2, and A.3), we obtain the formula for the distortion function:

$$g(u,v) = \left|(R/3R_0)(2/i)^{1/3}(1-s^2)^{1/6}(9-s^2)^{1/6}/s^{1/3}\right| \tag{A.4}$$

where

$s = sn(u+iv, 1/9)$ - coordinate on the $\mathbb{S}$-plane [2];

sn - elliptic sine function.

Figure A.3 shows the graph of the distortion function.

Using (A.3), we can compute the dimensions of the rectangle (Fig.3.1) on the

[2]The variable s is convenient for two reasons: first, the distortion function formula has a simpler form; second, the real axis of the $\mathbb{S}$-plane corresponds to boundary points of the computational cell

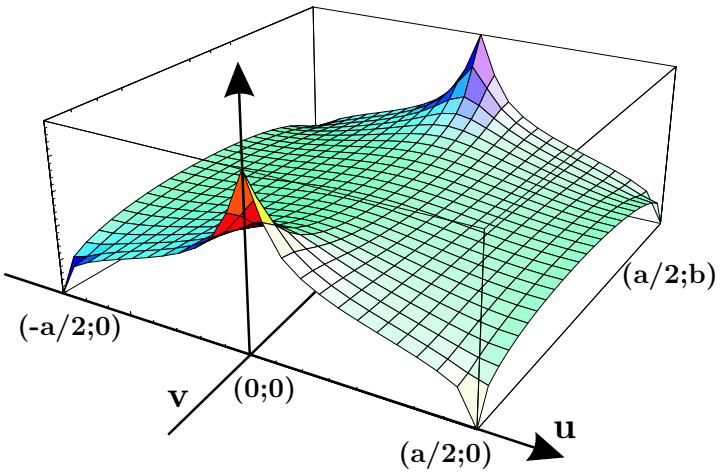


Figure A.3: Graph of the distortion function $g(u,v)$.

$\mathbb{W}$ plane:

$$a = 2\int_0^1 [(1-p^2)^{1/2}(1-p^2/9)^{1/2}]^{-1}\mathrm{d}p \cong 3.234773 \tag{A.5}$$
$$b = \Im\{\int_0^1 [(1-p^2)^{1/2}(1-p^2/9)^{1/2}]^{-1}\mathrm{d}p\} \cong 2.528626$$

These dimensions correspond to mapping a unit circle to a rectangle. To relate the rectangle size to the physical size of the hexagon, we use the area equality condition (3.15). This gives:

$$a \cong 1.052556h \tag{A.6}$$
$$b \cong 0.822783h$$

Figures A.4 and A.5 show the graphs of $g(\xi,0)$ and $\overline{g^2}(\xi)$. Note that $\lim_{u\to 0} g(u,0) = \lim_{u\to 0} g(u,b) = \infty$. Thus, while the conformal mapping doesn't completely eliminate singularities at the hexagon vertices that arise during transverse integration, since the integral of $g(u,0)$ over $[0,1]$ is bounded, this doesn't cause any problems.

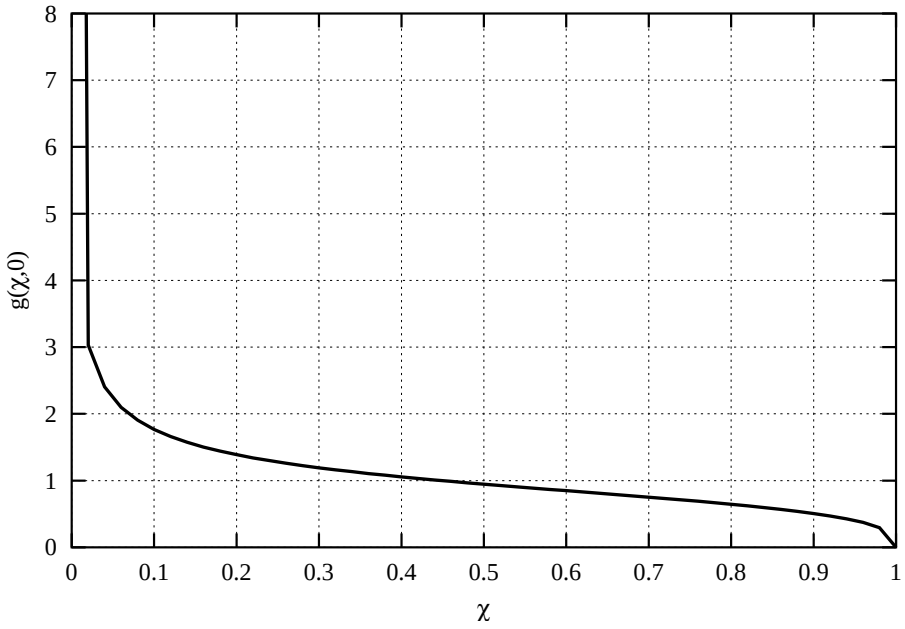


Figure A.4: Graph of the function $g(\xi, 0)$

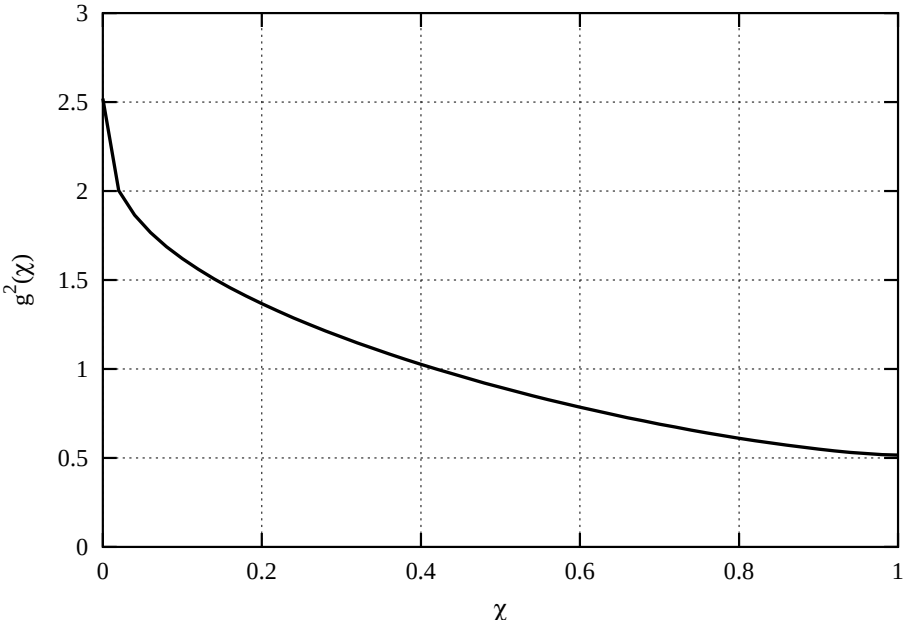


Figure A.5: Graph of the function $\overline{g^2}(\xi)$