CROSS SECTION GENERATION AND NEUTRONIC CALCULATIONS WITH NODAL METHODS

NODAL YÖNTEMLER İLE TESİR KESİTİ ÜRETİMİ VE NÖTRONİK HESAPLAMALAR

METİN KÖKSAL

A THESIS OF MASTER OF SCIENCE

Prepared in The NUCLEAR ENGINEERING DEPARTMENT

According To Regulations of

The Institute For Graduate Studies in Pure

And Applied Sciences of Hacettepe University

2006

To the Institute for Graduate Studies in Pure and Applied Sciences,

This work has been approved by the Graduate Commitee as partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** in the field of **NUCLEAR ENGINEERING**

Graduate Committee

Head _____

Assoc. Prof. Dr. Mehmet Tombakoğlu

Member _____

Dr. Ayhan Yılmazer

Member _____

Assist. Prof. Dr. S. Sinan Keskin

APPROVED

This thesis has been approved on / / 2006 by the Graduate Commitee established by the Board of the Institute

/ / 2006

Prof. Dr. Ahmet R. ÖZDURAL
HEAD OF THE INSTITUTE FOR GRADUATE STUDIES IN PURE AND APPLIED SCIENCES

CROSS SECTION GENERATION AND NEUTRONIC CALCULATIONS WITH NODAL METHODS

Metin Köksal

ABSTRACT

The objective of this thesis is to use analytical nodal method and Monte Carlo technique to obtain a hybrid method in order to generate neutron cross sections and diffusion coefficients used in diffusion type of calculations. The Monte Carlo code MCNP (Monte Carlo N-Particle) which is a general purpose Monte Carlo transport code developed in Los Alamos National Laboratory is used for neutronic calculations. The slab geometry model is generated and used in MCNP to obtain cell neutron fluxes and reaction rates as well as multiplication factor. These results were used for obtaining homogenized node cross sections for each node. Nodal diffusion equations, which use modified one group theory, for slab and cylindrical geometry are derived and utilized in the analysis to determine diffusion coefficients of each node. Since it is important to make fast and accurate calculation in nuclear reactor analysis, we developed a method to perform fast and reliable cross section generation process for full core diffusion calculation for nuclear reactor analysis.

Keywords: Nodal Methods, MCNP, neutronic calculations, cross-section generation.

Advisor: Assoc. Prof. Mehmet Tombakoğlu, Hacettepe University, Department of Nuclear Engineering, Nuclear Engineering Section

NODAL YÖNTEMLER İLE TESİR KESİTİ ÜRETİMİ VE NÖTRONİK HESAPLAMALAR

Metin Köksal

ÖZ

Bu tezin amacı, difüzyon hesaplamalarında kullanmak için nötron tesir kesitlerinin ve difüzyon katsayısının bulunması amacıyla analitik nodal ve Monte Carlo tekniklerinden karma bir metot elde etmektir. Nötronik hesapların yapılması için Los Alamos Ulusal Laboratuvarı'nda geliştirilen genel amaçlı Monte Carlo transport kodu MCNP (Monte Carlo N-Particle) kullanıldı. Geometry modeli yapıldı ve nötron akısı ve reaksiyon oranı bilgileriyle nötron çoğalma katsayısının elde edilmesi için MCNP kodu kullanıldı. Bu sonuçlar nodal kod için homojenleştirilmiş nod tesir kesitlerinin elde edilmesinde kullanıldı. Bu analizde değiştirilmiş bir grup teori kullanan nodal difüzyon denklemleri levha ve silindirik geometry için türetildi ve her bir nodun difüzyon katsayısının belirlenmesi analizinde kullanıldı. Nükleer reaktör analizinde hızlı ve doğru hesaplama önemli olduğundan nükleer reaktör analizinde tüm kor difüzyon hesaplamaları için hızlı ve güvenilir bir şekilde tesir kesiti üretimi metodu gelistirdik.

Anahtar Sözcükler: Nodal yöntemler, MCNP, nötronik hesaplamalar, tesir kesiti üretimi.

Danışman: Doç. Dr. Mehmet Tombakoğlu, Hacettepe Üniversitesi, Nükleer Enerji Mühendisliği Bölümü, Nükleer Enerji Mühendisliği Anabilim Dalı

ACKNOWLEDGMENTS

I would like to thank to my advisor Assoc. Prof. Mehmet Tombakoğlu for his helpful cooperation, guidance and constructive suggestions.

I also want to thank committee members Assist. Prof. Dr. S. Sinan Keskin and Dr. Ayhan Yılmazer.

Special thanks to my family for their support and patience throughout all my life and this work. Finally, I wish to thank all the members of Hacettepe University Nuclear Engineering Department.

Metin Köksal

Hacettepe University

September 2006

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Foreword

Of all modern technologies, nuclear reactor technology is unique in having sprung up full blown almost overnight. Only four years seperate the date of the discovery of fission (1938) and the date of the first chain reaction (1942). The first moderately powered reactor (Oak Ridge Graphite Reactor) went critical late in 1943; the first really high powered reactor started in 1944 at Hanford. Electricity was first produced from fission at the Experimental Breeder Reactor I in Idaho in 1951.

More than 400 nuclear power plants currently operate throughout the world, supplying about 16% of the world's electricity. It is well known that unlike fossil fuel plants, nuclear plants do not release carbon dioxide, sulfur, or nitrogen oxides into the environment. The world's increasing demand for energy, and the growing concerns for the environment, will favor energy sources with minimal environmental impact and competitive economics. Nuclear power plants can and should play a role in the development of energy supply in the future and need to meet the new goals and challenges, which the electricity energy market poses.

Nuclear fuel management plays a crucial role in meeting safety requirements, reducing the energy production cost and maximizing operating flexibility of nuclear power plants. There are two main categories in the nuclear fuel management. Out-

of-core fuel management involves activities which takes place prior to fuel irradiation called the front-end of fuel cycle and activities that involve operations on the highly radioactive spent fuel and low-level waste called back-end of nuclear fuel cycle. The main goal of in-core-fuel management activities is to achieve the design objectives (energy requirement, safety).

There are several factors related with performing in-core fuel management calculations. One of them is energy requirement and it depends on utility's demand. Once the energy demand is specified for a given period, the core reactivity calculations can be performed. Then the other unknowns, such as amount and enrichment of the fresh fuel, fraction of the depleted fuel to be removed, burnable poison requirements and loading pattern map must be determined. Such calculations are required to optimize in-core fuel cycle under some constraints to satisfy the utility's demand. To perform such calculations, spatial power distribution is needed under steady state operating conditions. Such a calculation is one of the main problems of nuclear engineering and is composed of two steps. The first step is cross section preparation as a function of the types of the assembly and burnable poisons. Once the homogenized cross sections characterizing each region in the core as a function of burn-up, fuel temperature and moderator temperature are generated, the full core spatial calculations are performed as a function of burn-up.

1.2 Nodal Method

The basic equation of diffusion theory is sufficiently accurate for the calculation of neutron densities in systems which are homogeneous, or nearly homogeneous, and which are so large that the radius of curvature of their boundaries is almost everywhere large as compared with the mean free path of the neutrons. This conditions are satisfied in large homogeneous reactors of the usual, rather regular shapes. On the other hand, small enriched reactors, whether homogeneous or heterogeneous, do not satisfy these conditions. For such systems transport theory provides a much more accurate description of the neutron distribution than diffusion theory.

The neutron distribution in a reactor core is one of the fundamental quantities required in the simulation and design of nuclear reactors. An exact description of this distribution (i.e. the angular neutron density) is provided by the Boltzmann transport equation (1.1).

$$\frac{1}{v}\frac{\partial}{\partial t}\varphi(r,E,\hat{\Omega},t)+\hat{\Omega}\cdot\nabla\varphi(r,E,\hat{\Omega},t)+\Sigma_t(r,E)\varphi(r,E,\hat{\Omega},t)$$
$$=\int_{4\pi}d\hat{\Omega}'\int_0^\infty dE'\Sigma_s(E'\to E,\hat{\Omega}'\to\hat{\Omega})\varphi(r,E',\hat{\Omega}',t)+s(r,E,\hat{\Omega},t) \quad (1.1)$$

where

v = Neutron speed (cm/s)

t = Time (s)

$r \in R^3$ = Space vector

$\Omega \in R^2$ = Angular direction vector

s = Source term (neutrons/cm^3/s)

The angular flux defined as

$$\varphi(r,E,\hat{\Omega},t)=vn(r,E,\hat{\Omega},t) \quad (1.2)$$

where

$$v=\left(\frac{2E}{m}\right)^{1/2} \quad (1.3)$$

and n is defined as

$n(r,E,\hat{\Omega},t)$ = expected number of neutrons in $dr, dE, d\hat{\Omega}$ around the phase space point $r, E, \hat{\Omega}$ at time t.

Unfortunately, this integrodifferential equation is extremely difficult to solve and although its numerical treatment has improved significantly over the past 30 years, it is often replaced with approximate models. The most popular of these is the P_N approximation, which correspond to an N^{th} order angular expansion of the angular neutron density, and is asymptotic to the transport equation. This expansion leads to a system of N first order PDEs, and in particular the P_1 approximation leads to the first order form of the diffusion equation. In general, the energy dependence of the angular

neutron density is discretized into groups such that a system of diffusion equations is obtained. These multi group diffusion equations have been the backbone of reactor modelling and continue to play a significant role. The diffusion model represents a significant simplification of the transport equation; however, fine scale variations in the coefficients in combination with the large physical size of the reactor core pose a complex computational problem. Yet the coarse scale properties of the solution (e.g. cell averages), that appear on a practical computational mesh, are significantly influenced by the fine scale structure of the coefficients. Problems containing two or more significantly different length scales arise frequently in physical applications and their treatment relies on various homogenization procedures. Specifically, the fine scale structure of the coefficients is averaged in some sense to define an homogenized diffusion problem in which the coefficients are piecewise smooth functions, typically constants, on the computational mesh. Unfortunately, the earliest treatments of the multigroup diffusion equations, which applied standart discritization tecniques to this homogenized problem, proved inadequate. Efforts to bridge this gap between computational resources and sophisticated modelling have produced a powerful class of discritization tecniques referred to as *nodal methods*.

Common to all *nodal methods* is their rigorous enforcement of cell balance, a characteristic motivated by the underlying transport equation. In fact, the first nodal methods were based on a clever hybrid approach that combined transport physics and diffusion theory. However, these methods are not consistent with the multigroup diffusion equations, and as a result have largely been superseded by the modern, consistent family of nodal discritizations. These particular methods are based on a transverse integration procedure that reduces the multidimensional diffusion equation to a coupled set of ODEs involving transverse averaged quantities. This not only facilitates the use of unknowns compatible with the popular homogenization tecniques (cell and edge moments in two dimensions) but has contributed to their enhancement. The treatment of these transverse integrated ODEs classifies the members of this family as either polynomial or analytic nodal methods.

Since 1970s significant progress was made in developing systematically de-

rived nodal methods. These modern or transverse integrated nodal methods can be derived in a systematic way and are capable of computing the eigenvalue and the node averaged fluxes with high accuracy on a very coarse mesh.

From mathematical viewpoint, nodal methods of reactor physics are special finite volume tecniques. To fully exploit the inherent advantages of the finite volume method for the solution of the neutron diffusion equation, accurate and efficient tecniques for the calculation of flux gradients on the surface of the finite volumes must be provided. This problem has been solved by the construction of a consistent transverse integrated one dimensional diffusion equation for each space direction. Several variants of the transverse integrated nodal diffusion theory methods can be distinguished by the methods used for solving the one dimensional diffusion equations [6].

Reactor physics calculations provide the basic information for in-core fuel management analysis. The major objective of these neutronic calculations is the prediction of core parameters such as reactivity, reaction rates (hence, power densities and burnup) and isotopic compositions. The level of accuracy required and thus the sophistication of the analysis methods to be used is problem specific. Well developed and very accurate neutronic computer codes are available to perform detailed analysis of reactor cores. However, the high cost and complexity associated with the use of these programs for multivariable optimization studies, particularly when the general degree of uncertainty inherent to the problem being modeled is muc larger than the accuracy provided, clearly points to the need to develop simpler and more cost efficient models based on analytical and empirical methods.

The Boltzman neutron transport equation is the fundamental relation for nuclear reactor analysis [1]. It is a conservation equation for the angular neutron density as a function of position, direction of motion and neutron energy, describing the interaction of neutrons with their environment. However, for real life situations, its solution is either too costly or impractical, and lower order approximations, with degree of accuracy compatible with the requirements of each specific situation, are used in reactor analysis. For most applications the main workhorse is the neutron diffusion approximation. In general, a further assumption of separability of space, time and spectrum

effects is also necessary. Under this assumption, the core is divided into regions having similar characteristics, for which few group constants are generated in independent spatially simlified computations, and then used in few group spatial calculations.

1.3 Cross Section Generation

The microscopic cross section (σ) is a property of a given nuclide; σ is the probability per nucleus that a neutron in the beam will interact with the nucleus; this probability is expressed in terms of an equivalent area that the neutron "sees". The macroscopic cross section (Σ) takes into account the number of those nuclides present ($\Sigma = N\sigma\ [cm^{-1}]$).

Our goal is to analyze nuclear reactors. In addition to the description of the physical processes, we need to come up with a mathematical form (a set of equations) that will help us to quntify what is happening in the reactor core. The equations we use wil contain mathematical descriptions of the rates at which neutrons interact with the nuclei. Thus, we must describe such interactions mathematically. One of the purpose of this thesis is to develop a computational method used for the criticality analysis of regular lattice configurations by solving these equations. According to solve these equations one must know the constants in these equations. The method will be developed in the framework of one and a half group nodal diffusion theory, and nodal parameters will be determined by resorting to continuous energy Monte Carlo tecniques. The main problem in generating the multi group homogenized constants is to define and calculate the diffusion coefficient.

For generating cross sections first step is to obtain problem independent group constants. By using a flat weighting function, the parameters are averaged on a fine energy group structure. Then in the second step constructed libraries are used as a source for group constants condensation into a coarser group structure using some rough approximation to the problem dependent neutron averaging spectrum as the weighting function. At this step, energy intervals are sufficiently fine so that local variations in the neutron spectrum can be disregarded. The criterion, which defines a group of problems for which a data set is valid, is the similarity in the smooth neutron

spectrum. Such spectrum is used for weighting in the cross section averaging process.

The problem dependent few group constants are the result of the final stage of the data reduction process, starting from the multi group data and using neutron transport methods. The number of groups varies and spatial homogenization is also performed. Equivalent diffusion equation parameters (macroscopic cross sections and diffusion constants) can be deduced. Such data are highly oriented. They are calculated on a case by case basis and are normally considered as an application of nuclear data. Such a simple flux and volume weighting procedure is valid when there is no leakage from the homogenized region. This approach introduces significant errors in determination of neutron flow among assemblies in a real reactor configurations. This errors are somewhat mitigated by use of assembly discontinuity factors for conventional reactor core analysis.

Unlike deterministic codes, it is not necessary to constitute a problem dependent cross section set for Monte Carlo codes. Since it simulates full core, spectrum is directly related to the model and it gives more accurate results.

According to find cross sections MCNP code used. MCNP code does not give cross section directly. It give reaction rates. Since MCNP does not have required options to obtain cross section results, the homogenized cross section were obtained by processing MCNP reaction rates and fluxes by volume-averaging. Two aspects have been considered in developing the method: the capability to provide accurate results (i.e., with a level of precision comparable to that of the deterministic transport theory or Monte Carlo methods) and the computational efficiency. The computational efficiency is addressed by developing the method in the framework of nodal diffusion theory, whereas the accuracy is addressed through the way in which the nodal parameters are generated. In this work, the nodal constants are precomputed by resorting to continuous energy Monte Carlo techniques.

1.4 Thesis Organization

There are four chapters in this thesis. Introduction to nodal methods and neutron cross section generation are given in this chapter (Chapter 1).

The nodal method that used in this thesis is given in Chapter 2. Modified one group theory is given also given. Derivation of nodal method for slab geometry is done and a program written for slab geometry nodal calculations described. Results for these calculations also given in this chapter. Derivation of cylindrical geometry nodal equations given and for comparison criticality condition for cylindrical system given.

In the chapter 3, basic concept and procedure for neutron cross section generation are given. The usage of nodal method for determining the neutron diffusion coefficient is described.

Finally, summary of the work done and conclusion are given in the chapter 4.

Chapter 2

NODAL CALCULATION

2.1 Introduction

There are many instances in nuclear reactor analysis in which one requires a full three dimensional calculation of the neutron flux, for example, in core fuel depletion or control rod ejection studies. Although a direct numerical solution of the diffusion equation can be performed on a modern digital computer, it is extremely expensive to do so, particularly when a series of such calculations would be required for parameter study. Accuracy in the finite difference solution requires that the mesh spacing must be smaller than the thermal group diffusion length[11], and a typical LWR core is about 200 thermal diffusion lengths in each of the three dimensions, which results in several million mesh points, hence several million simultaneous equations. It is desired a scheme for determining the three dimensional core flux distribution that avoids the large storage and execution time requirements of a direct finite difference treatment of the diffusion equation.

Such a scheme is provided by so called *nodal methods*. General idea is to decompose the reactor core into relatively large subregions or *node cells* in which the material composition and flux are assumed uniform (or at least treated in an average sense). One then attempts to determine the coupling coefficients characterizing node cell to node cell leakage and then to determine the node cell fluxes themselves. The

global flux distribution and the effective multiplication factor are then determined from a nodal calculation.

In this chapter, a general derivation of modified one group model is illustrated, and overview of nodal methods is provided. Then some calculations in 1-D are made using nodal methods code.

2.2 Modified One Group Model

The major objective of the reactor physics calculations is the prediction of core parameters such as reactivity, reaction rates (hence power densities and burnup) and isotopic compositions. The level of accuracy required and thus the sophistication of the analysis methods to be used is problem specific.
A modified one group theory method, known as the *one-and-one-half* group model, can be derived from the two group diffusion equations.

fast group:

$$-\nabla \cdot D_1 \nabla \phi_1 + \Sigma_{a1}\phi_1 + \Sigma_{12}\phi_1 - \frac{1}{\lambda}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2) = 0 \tag{2.1}$$

thermal group:

$$-\nabla \cdot D_2 \nabla \phi_2 + \Sigma_{a2}\phi_2 - \Sigma_{12}\phi_1 = 0 \tag{2.2}$$

where λ is the neutron multiplication factor, ν the average number of neutrons released per fission, D diffusion coefficient, Σ_a macroscopic absorption cross section, Σ_f macroscopic fission cross section, Σ_{12} the macroscopic downscattering cross section, and with 1 and 2 standing for fast and thermal groups, respectively.

The essential approximation of the one-and-one-half group model is that thermal neutrons are absorbed at the point of removal from the fast group, or, equivalently, that thermal leakage is neglected (i.e.: $\nabla^2\phi_2 = 0$). Thermal leakage is an order of magnitude smaller than fast leakage in LWRs. If we note that in a large LWR,

$D_2 \sim 0.5cm$, $B^2 \cong 10^{-4}cm^{-2}$, while $\Sigma_{a_2} \cong 0.1cm^{-1}$, then we find

$$\frac{-D_2\nabla^2\phi_2}{\Sigma_{a_2}\phi_2} = \frac{D_2B^2}{\Sigma_{a_2}} \cong 5 \times 10^{-4} \ll 1$$

Hence the neglect of $-D_2\nabla^2\phi_2$ may occasionally be valid for the investigation of problems in which thermal neutron diffusion does not have a significant reactivity effect [2].

Then Eq. (2.2) becomes:

$$\Sigma_{a2}\phi_2 = \Sigma_{12}\phi_1 \tag{2.3}$$

Substituting ϕ_2 from this expression in Eq. (2.1), and assuming D_1 constant over the reactor, gives

$$-D_1\nabla^2\phi_1 + \Sigma_{a1}\phi_1 + \Sigma_{12}\phi_1 - \frac{1}{\lambda}\left(\nu\Sigma_{f1} + \nu\Sigma_{f2}\frac{\Sigma_{12}}{\Sigma_{a2}}\right)\phi_1 = 0 \tag{2.4}$$

A very useful (and common) modification of this scheme which takes some account of both fast and thermal leakage within a one-group treatment is obtained by using *migration area* defined as

$$M^2 = D_1/(\Sigma_{a1} + \Sigma_{12})$$

and recalling that the local two group infinite medium neutron multiplication factor is:

$$k_\infty = \frac{\left(\nu\Sigma_{f1} + \nu\Sigma_{f2}\frac{\Sigma_{12}}{\Sigma_{a2}}\right)}{(\Sigma_{a1} + \Sigma_{12})} \tag{2.5}$$

Eq. (2.4) may be rearrange as:

$$\nabla^2\phi_1 + \left(\frac{k_\infty}{\lambda} - 1\right)\frac{1}{M^2}\phi_1 = 0 \tag{2.6}$$

Furthermore, defining the local reactivity,

$$\rho = 1 - 1/k_\infty \,,$$

and a modified eigenvalue, called here the *system (static) reactivity*,

$$\rho_s = 1 - 1/\lambda \,.$$

an equivalent expression for Eq. (2.6) is obtained, namely:

$$\nabla^2\phi_1 + \frac{(\rho-\rho_s)}{(1-\rho)}\frac{1}{M^2}\phi_1 = 0 \tag{2.7}$$

For a critical reactor (i.e.: $\lambda = 1, \rho_s = 0$) above equation become:

$$\nabla^2\phi_1 + \frac{\rho}{(1-\rho)}\frac{1}{M^2}\phi_1 = 0 \tag{2.8}$$

or, equivalently,

$$\nabla^2\phi_1 + \frac{(k_\infty - 1)}{M^2}\phi_1 = 0 \tag{2.9}$$

Buckling defined as follows:

$$B^2 = -\frac{(\rho-\rho_s)}{(1-\rho)M^2} \tag{2.10}$$

So the diffusion equation can be written as:

$$\nabla^2\phi - B^2\phi = 0 \tag{2.11}$$

It is important to notice that the system reactivity is a modified eigenvalue determining the core criticality. Equation 2.7 may be re-arranged and integrated over the core volume, V, to yield,

$$\rho_s = \frac{\int_V \frac{\rho}{1-\rho}\frac{D_1}{M^2}\phi_1 d\mathrm{v} + \int_V \nabla\cdot D_1\nabla\phi_1 d\mathrm{v}}{\int_V \frac{1}{1-\rho}\frac{D_1}{M^2}\phi_1 d\mathrm{v}} \tag{2.12}$$

From this equation system reactivity and so core criticality checked.

2.3 Nodal Method For Slab Geometry

Even after the local fuel pin, clad, coolant, and so on, heterogeneity is replaced by a homogenized representation, a reactor core remains a highly heterogeneous medium because of the intra assembly and assembly to assembly variation in fuel composition, burnable poisons, control rods, water channels, structure and so on. The mesh spacing in a conventional few group finite difference model of such a core is constrained by two requirements: (1) it must be sufficiently fine to represent the remaining spatial

heterogeneity adequately, and (2) it must be no larger than the shortest (thermal) group diffusion length in order to avoid numerical inaccuracy. A few group finite difference model that could adequately describe such a core might well have 10^5 to 10^6 unknowns (the fluxes in each group at each mesh point). The direct solution of such a problem, even in diffusion theory, remains a formidable computation.

Over the two past decades, efficient numerical techniques have been proposed to solve these equations accurately. Among these various methods, nodal schemes were the most attractive. They are fast and accurate which combine attractive features of the finite element method as well as the finite difference method. Nodal methods provide an efficient and accurate means of analysis whenever a reactor may be represented by regularly repeating, internally homogenized regions **(*nodes*)**.

Nodal methods characterize the global neutron flux distribution in terms of a small number of parameters in each of several large regions, or nodes, into which the reactor core is subdivided for this purpose. Such methods generally require detailed heterogeneous intranodal flux distributions to construct homogenized parameters for each of the many nodes into which a reactor core may be divided and to calculate coupling parameters that link the average flux solutions in adjacent nodes. The global average nodal fluxes must then be combined with the intranodal heterogeneous flux solutions if a heterogeneous flux distribution is required.

According to find analytical solution of the one-and-one-half group theory equation we made some assumption: the continuity of fluxes and currents at the nodal interfaces,the invariance of the fast group diffusion coefficient throughout the fueled nodes.

The one dimensional version of the one-and-one-half group equation was solved as follows. Each assembly was assumed to be a slab node, having width h_i. Subscript i shows coordinate of the node. Where i shows central node, i+1 indicates right node, and i-1 indicates the left node. There is two different solution; one for interior node, and one for peripheral node.

Eq.(2.11) is an elliptic differential equation having a unique solution within a closed boundary.

The solution to Eq.(2.11) is generically;

$$\phi_i(x) = A_i \cosh(B_i x) + G_i \sinh(B_i x) \tag{2.13}$$

By integrated this as:

$$\bar{\phi}_i = \int_{-h/2}^{h/2} \phi_i(x) dx \tag{2.14}$$

from this integrated flux of the node may be defined as:

$$\bar{\phi}_i = \left(\frac{2A_i}{B_i} \sinh(B_i h/2) \right) \tag{2.15}$$

Now the solution may be carried out for x direction, once the boundary conditions are specified. Where each node would be formally coupled to the surrounding ones. Here, the solution will be obtained for individual nodes under the assumption that the necessary quantities from the surrounding nodes ($\bar{\phi}_i$) are known. The objective is to solve for $\bar{\phi}_i$, as a function of the characteristics of the node i itself and its surrounding.

Boundary conditions (fluxes and currents are continuous at node interfaces) are written below.

For fluxes;

$$\phi_i(-h/2) = \phi_{i-1}(h/2) \tag{2.16}$$

$$\phi_i(h/2) = \phi_{i+1}(-h/2) \tag{2.17}$$

and for currents;

$$D_i \frac{d\phi_i(x)}{dx}\Big|_{x=-h/2} = D_{i-1} \frac{d\phi_{i-1}(x)}{dx}\Big|_{x=h/2} \tag{2.18}$$

$$D_i \frac{d\phi_i(x)}{dx}\Big|_{x=h/2} = D_{i+1} \frac{d\phi_{i+1}(x)}{dx}\Big|_{x=-h/2} \tag{2.19}$$

For interior nodes:

Using boundary conditions (fluxes and currents are continuous at node interfaces) and assuming as known the integrated fluxes in the neighboring nodes

$$\bar{\phi}_{i-1} = \left(\frac{2A_{i-1}}{B_{i-1}} \sinh(B_{i-1} h/2) \right) \tag{2.20}$$

and

$$\bar{\phi}_{i+1} = \left(\frac{2A_{i+1}}{B_{i+1}} \sinh(B_{i+1} h/2) \right) \tag{2.21}$$

The objective is to solve for the integrated flux

$$\bar{\phi}_i = \frac{2A_i}{B_i}\sinh(B_i h/2)$$

where the only unknown is A_i, which can be obtained from above equations and boundary conditions. The result for interior node integrated flux ϕ_i is reproduce as follows:

$$\frac{\bar{\phi}_i B_i^2 h_i^2}{2} = \frac{C_1 + C_2}{C_3 + C_4} \tag{2.22}$$

where the C_1, C_2, C_3, and C_4 are as follows:

$$C_1 = \bar{\phi}_{i-1} B_{i-1} h_{i-1} csch(B_{i-1}h_{i-1}) \left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i+1}h_{i+1}/2)}{D_{i+1}B_{i+1}h_{i+1}}\right] \tag{2.23}$$

$$C_2 = \bar{\phi}_{i+1} B_{i+1} h_{i+1} csch(B_{i+1}h_{i+1}) \left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i-1}h_{i-1}/2)}{D_{i-1}B_{i-1}h_{i-1}}\right] \tag{2.24}$$

$$C_3 = \left[\frac{\coth(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i-1}h_{i-1}/2)}{D_{i-1}B_{i-1}h_{i-1}}\right]\left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i+1}h_{i+1}/2)}{D_{i+1}B_{i+1}h_{i+1}}\right] \tag{2.25}$$

$$C_4 = \left[\frac{\coth(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i+1}h_{i+1}/2)}{D_{i+1}B_{i+1}h_{i+1}}\right]\left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i-1}h_{i-1}/2)}{D_{i-1}B_{i-1}h_{i-1}}\right] \tag{2.26}$$

For peripheral nodes:

Again solving for the x direction, the boundary conditions at the interior interfaces are the same as for the previous case. At the interface with the reflector, additional condition is imposed as the albedo boundary condition. We used a constant (α_n) for relation between flux and current in the reflector interface as follows:

$$J = \alpha_n \phi \tag{2.27}$$

This α_n is found from reflection coefficient or albedo, which is defined as the ratio between the current out of the reflecting region to the current into the reflecting region:

$$\alpha = \frac{J_{out}}{J_{in}} \tag{2.28}$$

For thin reflector, very few of the neutrons are reflected and hence the albedo is small. As the reflector becomes very thick, the albedo approach an asymptotic limit dependent only upon the material properties D and L as;

$$\alpha \rightarrow \alpha_{\infty} = \frac{1 - \frac{2D}{L}}{1 + \frac{2D}{L}} \tag{2.29}$$

For example, the albedos characterizing infinitely thick reflectors of graphite, water, and heavy water are 0.93, 0.82, and 0.97 respectively.

The albedo can be used to replaced the detailed solution in the reflecting region by an equivalent boundary condition at the edge of the core using Equation 2.28. That is, boundary condition on the flux in the previous region is just

$$\frac{1}{\phi} D \frac{d\phi}{dx}\Big|_{x=0} = -\frac{1}{2}\left(\frac{1-\alpha}{1+\alpha}\right) \tag{2.30}$$

Used in this manner, the albedo becomes a very useful device for obtaining boundary conditions for reactor core calculations [2].

From Eq. 2.27 and Eq. 2.30 the relation between α and α_n can be easily found as

$$\alpha_n = \frac{1}{2}\left(\frac{1-\alpha}{1+\alpha}\right) \tag{2.31}$$

So if the albedo known, α_n can be calculated easily. For example, if no neutron enter the system (vacuum boundary condition) $\alpha = 0$ and $\alpha_n = 0.5$.

From these the result for peripheral node integrated flux is obtained as follows:

$$\frac{\bar{\phi}_i B_i^2 h_i^2}{2} = \frac{E_1}{E_2 + E_3} \tag{2.32}$$

$$E_1 = \bar{\phi}_{i-1} B_{i-1} h_{i-1} csch(B_{i-1} h_{i-1}) \tag{2.33}$$

$$E_2 = \frac{\coth(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i-1} h_{i-1}/2)}{D_{i-1} B_{i-1} h_{i-1}} \tag{2.34}$$

$$E_3 = \frac{\left[\frac{\coth(B_i h_i/2)}{B_i h_i} + \frac{D_i}{\alpha_n h_i}\right]}{\left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i}{\alpha_n h_i}\right]} \left[\frac{\tanh(B_i h_i/2)}{B_i h_i} + \frac{D_i \tanh(B_{i-1} h_{i-1}/2)}{D_{i-1} B_{i-1} h_{i-1}}\right] \tag{2.35}$$

Using these equations nodal integrating fluxes are calculated.

2.4 One Dimensional Nodal Program

According to find criticality of the system (hence system reactivity) a program was written in the FORTRAN-77 [10] called SLABNOD given in the appendix of this thesis. This code, incorporating the nodal method illustrated in this chapter, which allows to solve 1-D criticality problems for reactor lattices composed of homogenous regions.

The inputs for program are as follows; geometry entry (number of region, number of material), maximum number of iteration, reactivity of the each node, migration area, diffusion coefficient, and an α_n number for boundary condition.

First of all , program initialize the nodal integrated flux values ($\bar{\phi}_i = 1$ for $i = 1 \ldots n_r$). Using peripheral and interior node equations find the $\bar{\phi}_i$ values. According to these fluxes system reactivity calculated using Equation 2.12. Then compare the iteration results flux and reactivity values with previous values for convergence. If these values converge stops the iteration and gives the results.Flowchart of the program is given in the Figure 2.1.

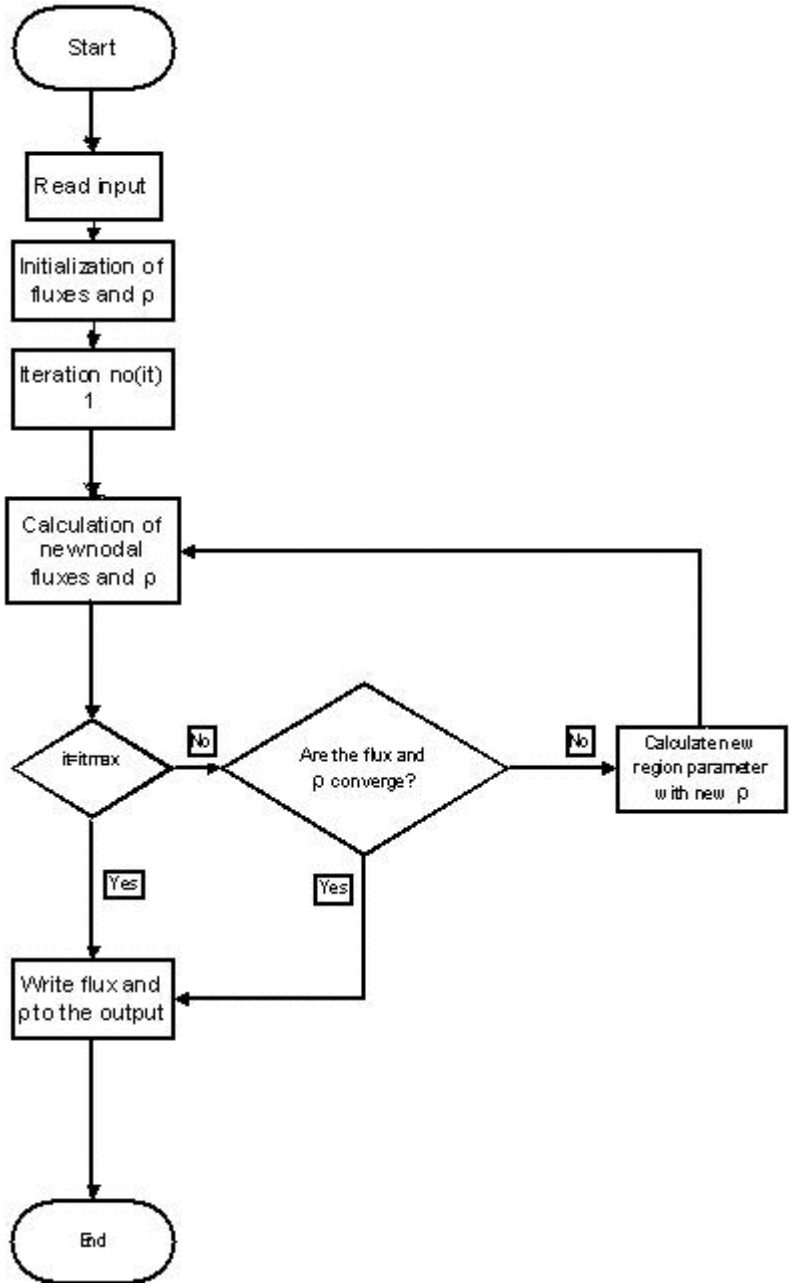

Figure 2.1: Flowchart of the slabnod program

2.5 Results For Slab Geometry Calculations

In this section the results of the application of the nodal method to the 1-D slab reactor problems are given. The code was tested by considering two seven-assembly configurations typical of a BWR core in 1-D as shown in Fig. 2.2.

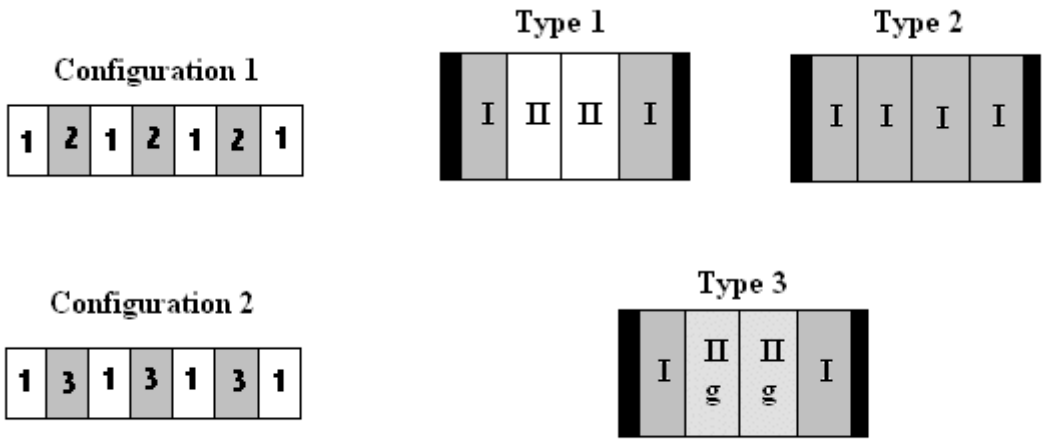

Figure 2.2: Core and assembly configuration.

A seven assembly core is made up of two unique alternating assemblies lined up in a one dimensional array. The type 1 assembly contains two enrichment zones where as the type 2 assembly has one enrichment zone with slightly smaller average enrichment. The type 3 assembly is the same as type 1 except that it contains gadolinium in fuel region II. The first configuration is a very simple case with relatively small changes across the core. The core and assembly configurations are shown in Fig.2.2 above. The corresponding cross sections and dimensions used for each region are shown in Table 2.1.

Using this parameters as input to the program we run the program for different cases and get eigenvalues.

The same calculations with different algorithm made by F.Rahnema and M.S.McKinley [3]. A comparison between our results and their results shown in Table 2. This shows the analysis of configuration 2, for different cases. In case 1 there is one assembly(only central one). We calculate and write the corresponding albedo (α). In

Table 2.1: Fuel assembly parameters

Atribute	Water	Fuel I	Fuel II	Fuel IIg
D(cm)	1.44	1.24	1.22	1.3
$\Sigma_a(cm^{-1})$	0.00205	0.0238	0.0284	0.075
$\nu\Sigma_f(cm^{-1})$	0.0	0.0314	0.0416	0.00844
Thickness	1.158	3.231	3.213	3.231

cases 2,3, and 4 we use 2,3,and 4 assembly and corresponding albedo.

Table 2.5 and 2.4 shows that our results are nearly same.

Table 2.2: Comparison of the eigenvalue

Case	Rahnema's results	Our results
1	0.8969	0.88969046
2	0.8968	0.89684498
3	0.89705	0.89702226
4	0.9022	0.902245998

Table 2.3: Comparison of the assembly integrated flux for configuration 1

Assembly	Rahnema's results	Our results
1	1.055	1.05503
2	0.984	0.9841
3	0.985	0.98464
4	0.950	0.95023

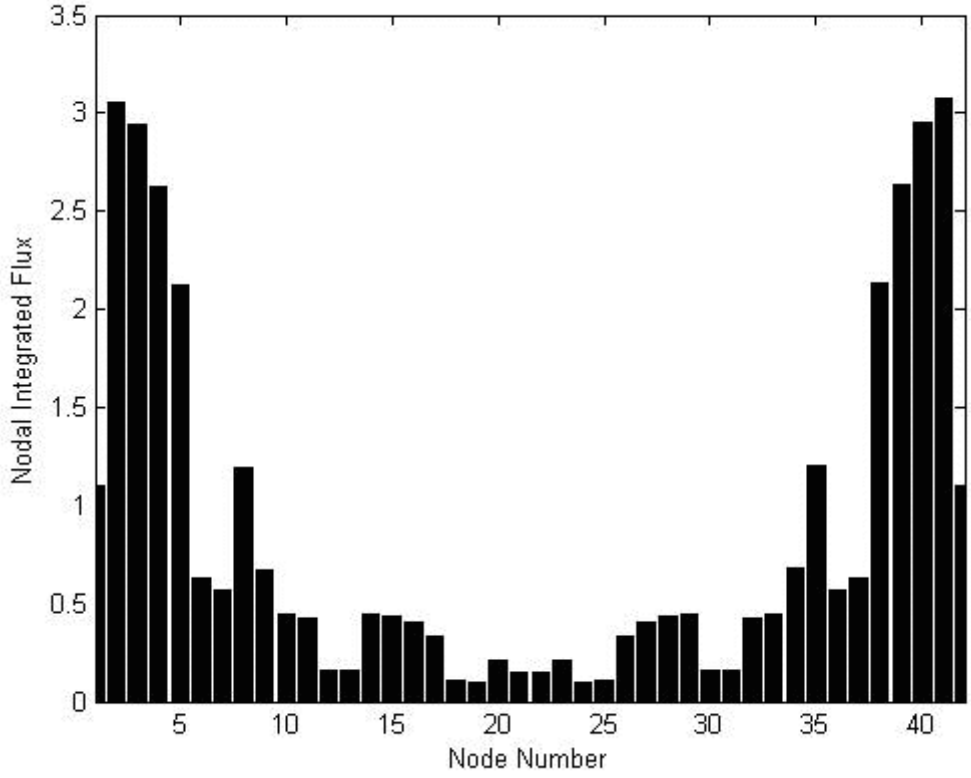

Figure 2.3: Nodal Flux Distrubution.

2.6 Cylindrical Geometry Nodal Method

Analysis of the cylindrical geometry nodal equations is similar to slab geometry solution. But the solution of the Equation 2.11 is as follows:

$$\phi_i(r) = A_i J_0(B_i r) + G_i Y_0(B_i r) \tag{2.36}$$

Since as a boundary condition flux must be finite at r=0, G_1 must be zero. So nodal flux in first node written as;

$$\phi_1(r) = A_1 J_0(B_1 r) \tag{2.37}$$

According to find integrated flux, flux equation integrate as:

$$\bar{\phi}_i = \int_{r_{i-1}}^{r_i} 2\pi r \phi_i(r) dr \tag{2.38}$$

Table 2.4: Comparison of the assembly integrated flux for configuration 2

Assembly	Rahnema's results	Our results
1	2.37	2.3711
2	0.658	0.65815
3	0.357	0.3564
4	0.173	0.1723

from this equation, integrated flux of the node i may be defined as:

$$\bar{\phi}_i = \frac{2\pi}{B_i}\Big[A_i(r_i J_1(B_i r_i) - r_{i-1}J_1(B_{i-1}r_{i-1})) + G_i(r_i Y_1(B_i r_i) - r_{i-1}Y_1(B_{i-1}r_{i-1}))\Big] \tag{2.39}$$

In a similar manner to slab geometry problem, derivation of the equations for peripheal node and interior nodes are done and a Fortran 77 code (CYLNOD) written for making these calculation. This code also given in the appendix.

The one dimensional version of the one-and-one-half group equation for cylindrical geometry was solved as follows. Each assembly was assumed to be a cylindrical section node, having width $r_i - r_{i-1}$ ($r_0 = 0$). Subscript i shows coordinate of the node. Where i shows central node, i+1 indicates right neighbor (next) node, and i-1 indicates the left neighbor (previous) node. There is three different solution; one for first (central) node, one for interior nodes, and one for peripheral node.

Now the solution may be carried out for r direction, once the boundary conditions are specified. Where each node would be formally coupled to the surrounding ones. Here, the solution will be obtained for individual nodes under the assumption that the necessary quantities from the surrounding nodes ($\bar{\phi}_i$) are known. The objective is to solve for $\bar{\phi}_i$, as a function of the characteristics of the node i itself and its surrounding. That is, A_i and G_i must be found.

For first node:

Boundary conditions (fluxes and currents are continuous at node interfaces and flux must be finite at the centre r=0) are written below. For fluxes;

Since Flux must be finite at r=0 , G_1 must be zero. Thus,

$$\phi_1(r) = A_1 J_0(B_1 r) \tag{2.40}$$

$$\phi_1(r_1) = \phi_2(r_1) \tag{2.41}$$

For currents;

$$D_1 \frac{d\phi_1(r)}{dr}\Big|_{r=r_1} = D_2 \frac{d\phi_2(r)}{dr}\Big|_{r=r_1} \tag{2.42}$$

The integrated fluxes of the neighbor node (node 2) is known as;

$$\bar{\phi}_2 = \frac{2\pi}{B_2}\Big[A_2(r_2 J_1(B_2 r_2) - r_1 J_1(B_2 r_1)) + G_2(r_2 Y_1(B_2 r_2) - r_1 Y_1(B_2 r_1))\Big] \tag{2.43}$$

$$A_1 J_0(B_1 r_1) = A_2 J_0(B_2 r_1) + G_2 Y_0(B_2 r_1) \tag{2.44}$$

$$D_1 B_1 A_1 J_1(B_1 r_1) = D_2 B_2 \Big[A_2 J_1(B_2 r_1) + G_2 Y_1(B_2 r_1)\Big] \tag{2.45}$$

From these three equation unknowns A_1, A_2, G_2 founds. So A_1 can be written as;

$$A_1 = \frac{E_1[J_1(B_2 r_1)Y_0(B_2 r_1) - J_0(B_2 r_1)Y_1(B_2 r_1)]}{\frac{D_1 B_1}{D_2 B_2} J_1(B_1 r_1) - J_0(B_1 r_1)[-E_2 J_1(B_2 r_1)]} \tag{2.46}$$

where E_1 and E_2 defined as follows;

$$E_1 = \bar{\phi}_2 \frac{B_2}{r_2 J_1(B_2 r_2) - r_1 J_1(B_2 r_1)} \tag{2.47}$$

$$E_2 = \frac{r_2 Y_1(B_2 r_2) - r_1 Y_1(B_2 r_1)}{r_2 J_1(B_2 r_2) - r_1 J_1(B_2 r_1)} \tag{2.48}$$

Since A_1 found, flux of the first node can be written;

$$\bar{\phi}_1 = \frac{1}{B_1}\Big[A_1 r_1 J_1(B_1 r_1)\Big] \tag{2.49}$$

For second node:

Boundary conditions (fluxes and currents are continuous at node interfaces) are written below. For fluxes;

$$\phi_i(r_{i-1}) = \phi_{i-1}(r_{i-1}) \tag{2.50}$$

$$\phi_i(r_i) = \phi_{i+1}(r_i) \tag{2.51}$$

and for currents;

$$D_i \frac{d\phi_i(r)}{dr}\Big|_{r=r_{i-1}} = D_{i-1} \frac{d\phi_{i-1}(r)}{dr}\Big|_{r=r_{i-1}} \tag{2.52}$$

$$D_i \frac{d\phi_i(r)}{dr}\Big|_{r=r_i} = D_{i+1} \frac{d\phi_{i+1}(r)}{dr}\Big|_{r=r_i} \tag{2.53}$$

And assuming that the integrated flux of the neighbor node (node i+1) is known as;

$$\bar{\phi}_{i+1} = \frac{2\pi}{B_{i+1}} \Big[A_{i+1}(r_{i+1} J_1(B_{i+1} r_{i+1}) - r_i J_1(B_{i+1} r_i)) + G_{i+1}(r_{i+1} Y_1(B_{i+1} r_{i+1}) - r_i Y_1(B_{i+1} r_i)) \Big] \tag{2.54}$$

$$A_i J_0(B_i r_{i-1}) + G_i Y_0(B_i r_{i-1}) = A_{i-1} J_0(B_{i-1} r_{i-1}) \tag{2.55}$$

$$A_i J_0(B_i r_i) + G_i Y_0(B_i r_i) = A_{i+1} J_0(B_{i+1} r_i) + G_{i+1} Y_0(B_{i+1} r_i) \tag{2.56}$$

$$D_i B_i \Big[A_i J_1(B_i r_{i-1}) + G_i Y_1(B_i r_{i-1}) \Big] = D_{i-1} B_{i-1} \Big[A_{i-1} J_1(B_{i-1} r_{i-1}) \Big] \tag{2.57}$$

$$D_i B_i \Big[A_i J_1(B_i r_i) + G_i Y_1(B_i r_i) \Big] = D_{i+1} B_{i+1} \Big[A_{i+1} J_1(B_{i+1} r_i) + G_{i+1} Y_1(B_{i+1} r_i) \Big] \tag{2.58}$$

There are five unknowns ($A_{i-1}, A_i, A_{i+1}, G_i, G_{i+1}$,) and five equations. Where i's are 2. According to these equations A_i and G_i are found.

$$S1 = J_0(B_{i+1} r_i) - \frac{r_{i+1} J_1(B_{i+1} r_{i+1}) - r_i J_1(B_{i+1} r_i)}{r_{i+1} Y_1(B_{i+1} r_{i+1}) - r_i Y_1(B_{i+1} r_i)} Y_0(B_{i+1} r_i) \tag{2.59}$$

$$S2 = J_1(B_{i+1} r_i) - \frac{r_{i+1} J_1(B_{i+1} r_{i+1}) - r_i J_1(B_{i+1} r_i)}{r_{i+1} Y_1(B_{i+1} r_{i+1}) - r_i Y_1(B_{i+1} r_i)} Y_1(B_{i+1} r_i) \tag{2.60}$$

$$S3 = \frac{S2}{S1} J_0(B_i r_i) - \frac{D_i B_i}{D_{i+1} B_{i+1}} J_1(B_i r_i) \tag{2.61}$$

$$S4 = \frac{D_i B_i}{D_{i+1} B_{i+1}} Y_1(B_i r_i) - \frac{S2}{S1} Y_0(B_i r_i) \tag{2.62}$$

$$S5 = \frac{\frac{S2}{S1} Y_0(B_{i+1} r_i) - Y_1(B_{i+1} r_i)}{r_{i+1} Y_1(B_{i+1} r_{i+1}) - r_i Y_1(B_{i+1} r_i)} \tag{2.63}$$

$$S6 = J_0(B_i r_{i-1}) \frac{J_1(B_{i-1} r_{i-1})}{J_0(B_{i-1} r_{i-1})} - \frac{D_i B_i}{D_{i-1} B_{i-1}} J_1(B_i r_{i-1}) \tag{2.64}$$

$$S7 = \frac{D_i B_i}{D_{i-1} B_{i-1}} Y_1(B_i r_{i-1}) - J_1(B_{i-1} r_{i-1}) \frac{Y_0(B_i r_{i-1})}{J_0(B_{i-1} r_{i-1})} \tag{2.65}$$

From these A_i and G_i found as follows;

$$A_i = \frac{S5}{S3 - \frac{S4S6}{S7}}\ \phi_{i+1}B_{i+1} \tag{2.66}$$

$$G_i = \frac{S5}{\frac{S3S7}{S6} - S4}\ \phi_{i+1}B_{i+1} \tag{2.67}$$

Using A_i and G_i expressions integrated flux of node i found;

$$\bar{\phi}_i = \tfrac{1}{B_i}\Big[A_i(r_iJ_1(B_ir_i) - r_{i-1}J_1(B_ir_{i-1})) + G_i(r_iY_1(B_ir_i) - r_{i-1}Y_1(B_ir_{i-1}))\Big] \tag{2.68}$$

For interior nodes:

Boundary conditions (fluxes and currents are continuous at node interfaces) are written below. For fluxes;

$$\phi_i(r_{i-1}) = \phi_{i-1}(r_{i-1}) \tag{2.69}$$

$$\phi_i(r_i) = \phi_{i+1}(r_i) \tag{2.70}$$

and for currents;

$$D_i\frac{d\phi_i(r)}{dr}\Big|_{r=r_{i-1}} = D_{i-1}\frac{d\phi_{i-1}(r)}{dr}\Big|_{r=r_{i-1}} \tag{2.71}$$

$$D_i\frac{d\phi_i(r)}{dr}\Big|_{r=r_i} = D_{i+1}\frac{d\phi_{i+1}(r)}{dr}\Big|_{r=r_i} \tag{2.72}$$

And assuming that the integrated fluxes of the neighbor nodes (node i-1 and i+1) are known as;

$$\bar{\phi}_{i-1} = \frac{2\pi}{B_{i-1}}\Big[A_{i-1}(r_{i-1}J_1(B_{i-1}r_{i-1}) - r_{i-2}J_1(B_{i-1}r_{i-2})) + G_{i-1}(r_{i-1}Y_1(B_{i-1}r_{i-1}) - r_{i-2}Y_1(B_{i-1}r_{i-2}))\Big] \tag{2.73}$$

$$\bar{\phi}_{i+1} = \frac{2\pi}{B_{i+1}}\Big[A_{i+1}(r_{i+1}J_1(B_{i+1}r_{i+1}) - r_iJ_1(B_{i+1}r_i)) + G_{i+1}(r_{i+1}Y_1(B_{i+1}r_{i+1}) - r_iY_1(B_{i+1}r_i))\Big] \tag{2.74}$$

$$A_iJ_0(B_ir_{i-1}) + G_iY_0(B_ir_{i-1}) = A_{i-1}J_0(B_{i-1}r_{i-1}) + G_{i-1}Y_0(B_{i-1}r_{i-1}) \tag{2.75}$$

$$A_iJ_0(B_ir_i) + G_iY_0(B_ir_i) = A_{i+1}J_0(B_{i+1}r_i) + G_{i+1}Y_0(B_{i+1}r_i) \tag{2.76}$$

$$D_iB_i\Big[A_iJ_1(B_ir_{i-1})+G_iY_1(B_ir_{i-1})\Big]=D_{i-1}B_{i-1}\Big[A_{i-1}J_1(B_{i-1}r_{i-1})+G_{i-1}Y_1(B_{i-1}r_{i-1})\Big] \quad (2.77)$$

$$D_iB_i\Big[A_iJ_1(B_ir_i)+G_iY_1(B_ir_i)\Big]= D_{i+1}B_{i+1}\Big[A_{i+1}J_1(B_{i+1}r_i)+G_{i+1}Y_1(B_{i+1}r_i)\Big] \quad (2.78)$$

There are six unknowns ($A_{i-1}, A_i, A_{i+1}, G_{i-1}, G_i, G_{i+1}$,) and six equations. Where i's are greater than 1 and less than the number of nodes. According to these equations A_i and G_i are found.

$$A_i=\frac{\epsilon_1\epsilon_2-\epsilon_3\epsilon_4}{\epsilon_5\epsilon_2-\epsilon_3\epsilon_6} \quad (2.79)$$

$$G_i=\frac{\epsilon_5\epsilon_4-\epsilon_1\epsilon_6}{\epsilon_5\epsilon_2-\epsilon_3\epsilon_6} \quad (2.80)$$

Where ϵ_i's are as follows;

$$\epsilon_1=\tfrac{D_{i-1}B_{i-1}}{D_iB_i}F_1\Big[J_1(B_{i-1}r_{i-1})Y_0(B_{i-1}r_{i-1})-J_0(B_{i-1}r_{i-1})Y_1(B_{i-1}r_{i-1})\Big] \quad (2.81)$$

$$\epsilon_2=Y_1(B_ir_i)\Big[Y_0(B_{i+1}r_i)-F_4J_0(B_{i+1}r_i)\Big]-\tfrac{D_{i+1}B_{i+1}}{D_iB_i}\Big[Y_1(B_{i+1}r_i)-F_4J_1(B_{i+1}r_i)\Big]Y_0(B_ir_i) \quad (2.82)$$

$$\epsilon_3=Y_1(B_ir_{i-1})\Big[Y_0(B_{i-1}r_{i-1})-F_2J_0(B_{i-1}r_{i-1})\Big]- \tfrac{D_{i-1}B_{i-1}}{D_iB_i}\Big[Y_1(B_{i-1}r_{i-1})-F_2J_1(B_{i-1}r_{i-1})\Big]Y_0(B_ir_{i-1}) \quad (2.83)$$

$$\epsilon_4=\tfrac{D_{i+1}B_{i+1}}{D_iB_i}F_3\Big[J_1(B_{i+1}r_i)Y_0(B_{i+1}r_i)-J_0(B_{i+1}r_i)Y_1(B_{i+1}r_i)\Big] \quad (2.84)$$

$$\epsilon_5=J_1(B_ir_{i-1})\Big[Y_0(B_{i-1}r_{i-1})-F_2J_0(B_{i-1}r_{i-1})\Big]- \tfrac{D_{i-1}B_{i-1}}{D_iB_i}\Big[Y_1(B_{i-1}r_{i-1})-F_2J_1(B_{i-1}r_{i-1})\Big]J_0(B_ir_{i-1}) \quad (2.85)$$

$$\epsilon_6=J_1(B_ir_i)\Big[Y_0(B_{i+1}r_i)-F_4J_0(B_{i+1}r_i)\Big]- \frac{D_{i+1}B_{i+1}}{D_iB_i}\Big[Y_1(B_{i+1}r_i)-F_4J_1(B_{i+1}r_i)\Big]J_0(B_ir_i) \quad (2.86)$$

The F values written as;

$$F_1 = \bar{\phi}_{i-1} \frac{B_{i-1}}{r_{i-1}J_1(B_{i-1}r_{i-1}) - r_{i-2}J_1(B_{i-1}r_{i-2})} \tag{2.87}$$

$$F_2 = \frac{r_{i-1}Y_1(B_{i-1}r_{i-1}) - r_{i-2}Y_1(B_{i-1}r_{i-2})}{r_{i-1}J_1(B_{i-1}r_{i-1}) - r_{i-2}J_1(B_{i-1}r_{i-2})} \tag{2.88}$$

$$F_3 = \bar{\phi}_{i+1} \frac{B_{i+1}}{r_{i+1}J_1(B_{i+1}r_{i+1}) - r_iJ_1(B_{i+1}r_i)} \tag{2.89}$$

$$F_4 = \frac{r_{i+1}Y_1(B_{i+1}r_{i+1}) - r_iY_1(B_{i+1}r_i)}{r_{i+1}J_1(B_{i+1}r_{i+1}) - r_iJ_1(B_{i+1}r_i)} \tag{2.90}$$

Using A_i and G_i expressions integrated flux of node i found;

$$\bar{\phi}_i = \tfrac{1}{B_i}\Big[A_i(r_iJ_1(B_ir_i) - r_{i-1}J_1(B_ir_{i-1})) + G_i(r_iY_1(B_ir_i) - r_{i-1}Y_1(B_ir_{i-1}))\Big] \tag{2.91}$$

For last node:

Boundary conditions at the interior interface are the same as for the previous case. At the interface with the reflector, additional conditions are imposed: the flux in addition to satisfying the continuity requirement at the interface, is assumed to vanish at the distance R into the reflector. Solution for the flux in the reflector is given by

$$\phi_r(r) = A_rJ_0(B_rr) + G_rY_0(B_rr) \tag{2.92}$$

Since ϕ_r=0 at R, $G_r = -A_rJ_0(B_rR)/Y_0(B_rR)$ so

$$\phi_r(r) = A_r\Big[J_0(B_rr) - Y_0(B_rr)J_0(B_rR)/Y_0(B_rR)\Big] \tag{2.93}$$

Accordingly, the equations become for i=number of region (n_r)

$$\phi_i(r_{i-1}) = \phi_{i-1}(r_{i-1}) \tag{2.94}$$

$$\phi_i(r_i) = \phi_r(r_i) \tag{2.95}$$

$$D_i\frac{d\phi_i(r)}{dr}\Big|_{r=r_{i-1}} = D_{i-1}\frac{d\phi_{i-1}(r)}{dr}\Big|_{r=r_{i-1}} \tag{2.96}$$

$$D_i\frac{d\phi_i(r)}{dr}\Big|_{r=r_i} = D_r\frac{d\phi_r(r)}{dr}\Big|_{r=r_i} \tag{2.97}$$

$$\bar{\phi}_{i-1} = \frac{2\pi}{B_{i-1}}\Big[A_{i-1}(r_{i-1}J_1(B_{i-1}r_{i-1}) - r_{i-2}J_1(B_{i-1}r_{i-2})) + G_{i-1}(r_{i-1}Y_1(B_{i-1}r_{i-1}) - r_{i-2}Y_1(B_{i-1}r_{i-2}))\Big] \tag{2.98}$$

Last condition is simply written as $J = \alpha\phi$ at surface, that is

$$2\pi r_i D_r B_r \Big[J_1(B_r r_i) - Y_1(B_r r_i) J_0(B_r R)/Y_0(B_r R) \Big] = \alpha \Big[J_0(B_r r_i) - Y_0(B_r r_i) J_0(B_r R)/Y_0(B_r R) \Big] \tag{2.99}$$

According to these A_i and G_i for last node found

$$A_i = \frac{S_4 \frac{\phi_{i-1} B_{i-1}}{2\pi K_1} \Big[J_1(B_{i-1} r_{i-1}) - \frac{T_2}{T_1} J_0(B_{i-1} r_{i-1}) \Big]}{S_1 S_4 + S_2 S_3} \tag{2.100}$$

$$G_i = \frac{S_3 \frac{\phi_{i-1} B_{i-1}}{2\pi K_1} \Big[J_1(B_{i-1} r_{i-1}) - \frac{T_2}{T_1} J_0(B_{i-1} r_{i-1}) \Big]}{S_1 S_4 + S_2 S_3} \tag{2.101}$$

Where S's, T's, and K's as follows

$$K_1 = r_{i-1} J_1(B_{i-1} r_{i-1}) - r_{i-2} J_1(B_{i-1} r_{i-2}) \tag{2.102}$$

$$K_2 = r_{i-1} Y_1(B_{i-1} r_{i-1}) - r_{i-2} Y_1(B_{i-1} r_{i-2}) \tag{2.103}$$

$$T_1 = Y_0(B_{i-1} r_{i-1}) - \frac{K_2}{K_1} J_0(B_{i-1} r_{i-1}) \tag{2.104}$$

$$T_2 = Y_1(B_{i-1} r_{i-1}) - \frac{K_2}{K_1} J_1(B_{i-1} r_{i-1}) \tag{2.105}$$

$$S_1 = \frac{D_i B_i}{D_{i-1} B_{i-1}} J_1(B_i r_{i-1}) - \frac{T_2}{T_1} J_0(B_i r_{i-1}) \tag{2.106}$$

$$S_2 = \frac{D_i B_i}{D_{i-1} B_{i-1}} Y_1(B_i r_{i-1}) - \frac{T_2}{T_1} Y_0(B_i r_{i-1}) \tag{2.107}$$

$$S_3 = D_i B_i J_1(B_i r_i) - \frac{\alpha}{2\pi r_i} J_0(B_i r_i) \tag{2.108}$$

$$S_4 = \frac{\alpha}{2\pi r_i} Y_0(B_i r_i) - D_i B_i Y_1(B_i r_i) \tag{2.109}$$

Using A_i and G_i expressions integrated flux of the last node can be calculated as;

$$\bar{\phi}_i = \frac{2\pi}{B_i} \Big[A_i (r_i J_1(B_i r_i) - r_{i-1} J_1(B_i r_{i-1})) + G_i (r_i Y_1(B_i r_i) - r_{i-1} Y_1(B_i r_{i-1})) \Big] \tag{2.110}$$

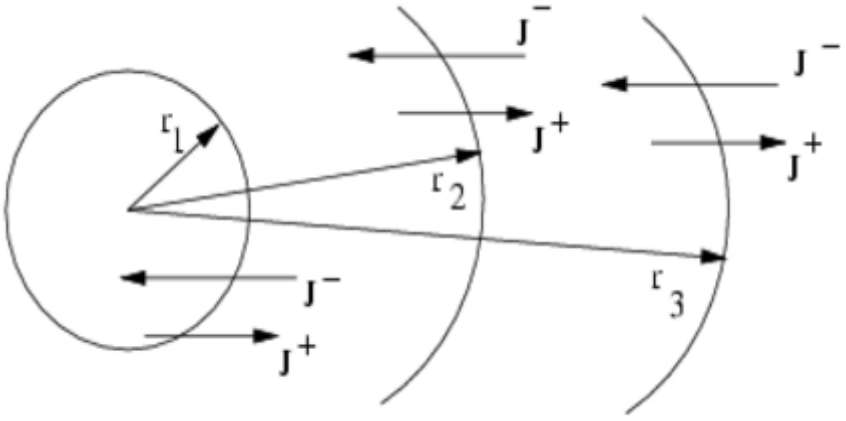

Figure 2.4: Geometry of the three region system

2.7 Criticality Condition For Cylindrical System

According to make cylnod code verification we derive analytical solution of the three region system shown in Figure 2.4. The first and the last nodes are reflecting region and second region is multiplying medium.

$$\phi_1(r) = A_1 I_0(K_1 r) \tag{2.111}$$

$$\phi_3(r) = A_3 I_0(K_3 r) + G_3 K_0(K_3 r) \tag{2.112}$$

$$J^+ = \frac{1}{4}\left[\phi_1 - 2D_1 \frac{d\phi_1}{dr}\right]_{r=r_1} \tag{2.113}$$

$$J^- = \frac{1}{4}\left[\phi_1 + 2D_1 \frac{d\phi_1}{dr}\right]_{r=r_1} \tag{2.114}$$

$$\beta_1 = \frac{J^-}{J^+} \tag{2.115}$$

So

$$\beta_1 = \frac{\left[\phi_1 + 2D_1 \frac{d\phi_1}{dr}\right]_{r=r_1}}{\left[\phi_1 - 2D_1 \frac{d\phi_1}{dr}\right]_{r=r_1}} \tag{2.116}$$

Hence

$$\beta_1 = \frac{I_0(K_1 r_1) + 2D_1 K_1 I_1(K_1 r_1)}{I_0(K_1 r_1) - 2D_1 K_1 I_1(K_1 r_1)} \tag{2.117}$$

There is vacuum boundary at $r = r_3$. So $J^-(r_3) = 0$.

$$\frac{1}{4}\left[\phi_3 + 2D_3\frac{d\phi_3}{dr}\right]_{r=r_3} = 0 \tag{2.118}$$

Using Equation 2.122 in above equation give relation between A_3 and G_3 as follows;

$$G_3 = -A_3\frac{I_0(K_3r_3) + 2D_3K_3I_1(K_3r_3)}{K_0(K_3r_3) - 2D_3K_3K_1(K_3r_3)} \tag{2.119}$$

Define a coefficient γ, such as $G_3 = \gamma A_3$.

Now, write β_2 at $r = r_2$ for region three.

$$\beta_2 = \frac{A_3[I_0(K_3r_2) + 2D_3K_3I_1(K_3r_2)] + G_3[K_0(K_3r_2) - 2D_3K_3K_1(K_3r_2)]}{A_3[I_0(K_3r_2) - 2D_3K_3I_1(K_3r_2)] + G_3[K_0(K_3r_2) + 2D_3K_3K_1(K_3r_2)]} \tag{2.120}$$

Since $G_3 = \gamma A_3$;

$$\beta_2 = \frac{[I_0(K_3r_2) + 2D_3K_3I_1(K_3r_2)] + \gamma[K_0(K_3r_2) - 2D_3K_3K_1(K_3r_2)]}{[I_0(K_3r_2) - 2D_3K_3I_1(K_3r_2)] + \gamma[K_0(K_3r_2) + 2D_3K_3K_1(K_3r_2)]} \tag{2.121}$$

For multiplying region (region two) flux written as;

$$\phi_2(r) = A_2J_0(K_2r) + G_2Y_0(K_2r) \tag{2.122}$$

If we write β_1 and β_2 for multiplying region at $r = r_1$ and $r = r_2$;

$$\beta_1 = \frac{A_2[J_0(K_2r_1) - 2D_2K_2J_1(K_2r_1)] + G_2[Y_0(K_2r_1) - 2D_2K_2Y_1(K_2r_1)]}{A_2[J_0(K_2r_1) + 2D_2K_2J_1(K_2r_1)] + G_2[Y_0(K_2r_1) + 2D_2K_2Y_1(K_2r_1)]} \tag{2.123}$$

$$\beta_2 = \frac{A_2[J_0(K_2r_2) - 2D_2K_2J_1(K_2r_2)] + G_2[Y_0(K_2r_2) - 2D_2K_2Y_1(K_2r_2)]}{A_2[J_0(K_2r_2) + 2D_2K_2J_1(K_2r_2)] + G_2[Y_0(K_2r_2) + 2D_2K_2Y_1(K_2r_2)]} \tag{2.124}$$

Manipulating these two equation cancels out A_2 and G_2 terms so we get criticality condition as;

$$\begin{aligned}\frac{(1-\beta_2)J_0(K_2r_2) - (1+\beta_2)2D_2K_2J_1(K_2r_2)}{(\beta_2-1)Y_0(K_2r_2) + (1+\beta_2)2D_2K_2Y_1(K_2r_2)} = \\ \frac{(1-\beta_1)J_0(K_2r_1) - (1+\beta_1)2D_2K_2J_1(K_2r_1)}{(\beta_1-1)Y_0(K_2r_1) + (1+\beta_1)2D_2K_2Y_1(K_2r_1)}\end{aligned} \tag{2.125}$$

β_1 known (from eqn. 2.117) as a function of first region parameters. β_2 known (from eqn. 2.121) as a function of third region parameters. So in the above equation K_2 is the only unknown and can be found.

If we take $\beta_1 = 1$ and $\beta_2 = 0$, this represents the homogeneous cylinder case. Using these β values one can make different calculations. Using this criticality equation system reactivity and so multiplication factor and shape of the flux distribution can be found.

Sample homogeneous cylinder flux distribution taken from cylnod program given in the Figure 2.5.

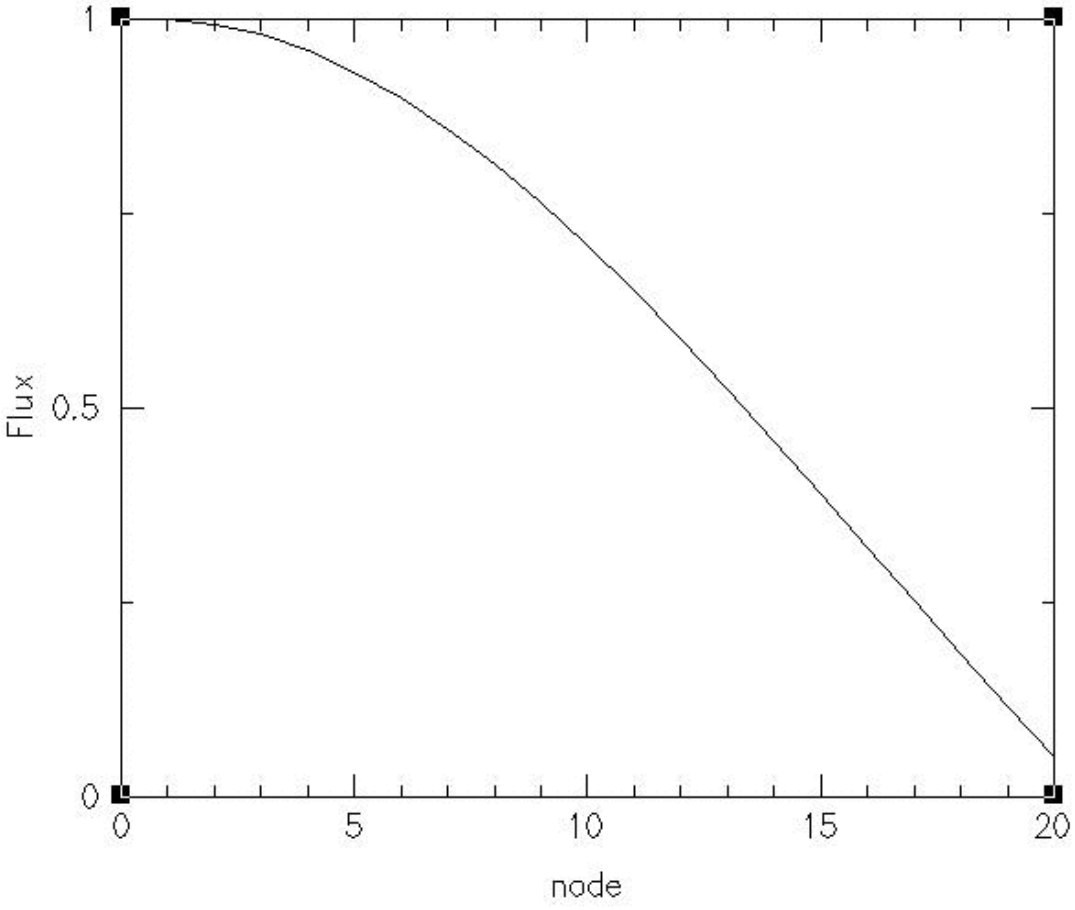

Figure 2.5: Flux distrubution in a cylinder

Chapter 3

CROSS SECTION GENERATION

3.1 Introduction

Monte Carlo methods of performing radiation transport calculations are heavily used in many different applications. However, despite their prevalence, Monte Carlo codes do not eliminate the need for other methods of analysis like discrete ordinates transport codes or even diffusion theory codes. For example, current Monte Carlo codes are not capable of performing transient analysis or continuous energy adjoint calculations.

One of the primary difficulties, however, in using non continuous energy methods is the need for reliable multigroup cross sections that have been collapsed over an appropriate flux spectrum. Therefore, we use Monte Carlo code MCNP to generate multigroup cross section. This will make it possible for the user to take advantage of the accuracy and ease of problem definition provided by MCNP while generating multi group cross sections for other applications such as nodal codes.

The calculation of multigroup cross section for use in various reactor physics codes is a task that can be rather difficult to perform. The user must calculate an appropriate flux spectrum which is used for weighting and collapsing the continuous energy or fine group cross sections. Typically, during cross section generation, the user is not able to represent accurately the geometry of the problem being analyzed. One and two dimensional approximations have to be made in order to calculate the flux

spectrum which will be used for cross section weighting. These approximations, if made carefully, are adequate to obtain correct multigroup cross section. However, the effort involved in generating these cross sections is still substantial.

Monte Carlo methods offer the user a great advantage in both physical accuracy and ease of problem setup. MCNP [4] uses continuous energy cross section data for the radiation transport. It has a full three dimensional modelling capability with a large number of user definable tallies and source definitions. Basic theory of the Monte Carlo methods and MCNP is given in the appendix.

3.2 Basic Concepts

The two group diffusion model can be used to demonstrate a number of the various applications of the multigroup formalism. For example, one frequently wishes to generate the group constants for a few group calculation using the neutron spectrum generated by a many group calculation. Such a procedure is known as *group collapsing*, since it expresses few group constants in terms of many group constants. Equation below is a general simplified definition of a multigroup cross section collapsed from continuous energy cross section data.

$$\Sigma_g = \frac{\int_{E_g}^{E_{g-1}} \Sigma(E)\phi(E)dE}{\int_{E_g}^{E_{g-1}} \phi(E)dE} \tag{3.1}$$

Σ_g = group cross section
$\Sigma(E)$ = macroscopic cross section as a function of energy
$\phi(E)$ = neutron flux as a function of energy

To illustrate group collapsing we can write expression for the one group constants in terms of two group constants.

$$\Sigma_a = \frac{\int_{E_2}^{E_0} \Sigma(E)\phi(E)dE}{\int_{E_2}^{E_0} \phi(E)dE} \tag{3.2}$$

For example one group diffusion coefficient can easily calculated from two group data

as;

$$D = \frac{D_1\phi_1 + D_2\phi_2}{\phi_1 + \phi_2} \tag{3.3}$$

In continuous energy Monte Carlo there are many different types of reactions that occur and therefore many different types of cross sections. However, in nodal methods only a limited number of cross sections are needed. For example total, absorbtion, and fission cross sections.

Instead of calculating group cross section in a single material, one may wish to calculate homogenized group cross sections over a region of heterogeneous materials. If there are two region system as shown in Figure 3.1 homogenized cross sections may be found as follows;

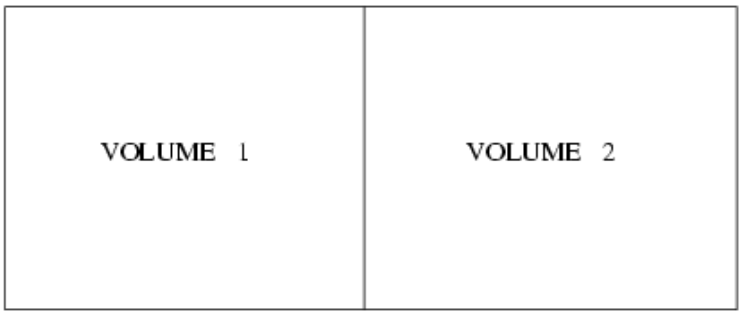

Figure 3.1: Heterogenous two region system

$$\phi_g = \frac{V^1\phi_g^1 + V^2\phi_g^2}{V^1 + V^2} \tag{3.4}$$

From this equation the cell averaged cross section can easily be derived and

$$\Sigma_g = \frac{V^1\Sigma_g^1\phi_g^1 + V^2\Sigma_g^2\phi_g^2}{V^1\phi_g^1 + V^2\phi_g^2} \tag{3.5}$$

Generating multigroup cross sections is a two step process. In step one, the flux and the various reacation rates are tallied during particle transport. The second step, which is performed after the calculation has been completed, is to divide the reaction rates by the flux, thereby calculating the macroscopic multigroup cross section.

3.3 Procedure

MCNP input preparation involves two parts: the model definition and preperation of tally cards. After the model is generated, it is used in seperate ways for cross section generation and tallying of the surface flux and currents. This section discusses the tallying process related to homogenized cross section generation part.

To tally the required data for homogenization, only the F4 type tally cards of MCNP are used [4]. Witout any multipliers, the F4 cards provide the track length estimate of MCNP cell fluxes. For reaction rate tallies the atom density data required. By using these atom densities and multipliers for absorption, total, and fission cross sections, and for $\nu\Sigma_f$ reaction rate tally cards prepared. To tally for axial nodes over portions of the MCNP cells, tally segmentation cards are used.

MCNP calculates cell volumes and surface areas as one of the first stages of the run, but it cannot calculate the volumes and areas of asymmetric, nonpolyhedron, or infinite cells. When the cells in the model are asymmetric, the volumes of this cells are found by hand calculations if the geometry is simple.

The input files for MCNP runs are ready after the model implementation and preparation of the tally cards. In order to find cross section we used geometry shown in Figure 3.2. And input prepared for this calculation given in the appendix. The boundary condition for left side is reflecting boundary condition.

The node homogenized cross sections are obtained by flux weighted volume averaging using the computer programs written to process the MCNP tally output files. As known the flux weighted volume averaging formula for homogenization can be given as the following:

$$\Sigma_g^j = \frac{\sum_i R_g^i V^{(i,j)}}{\sum_i \phi_g^{(i,j)} V^{(i,j)}} \tag{3.6}$$

where,

Σ_g^j = homogenized cross section for energy group g and node j

R_g^i = MCNP reaction rate per unit volume for cell i

$V^{(i,j)}$ = volume of the portion of cell i that is in node j

$\phi_g^{(i,j)}$ = MCNP flux density for energy group g and the portion of cell i in node j.

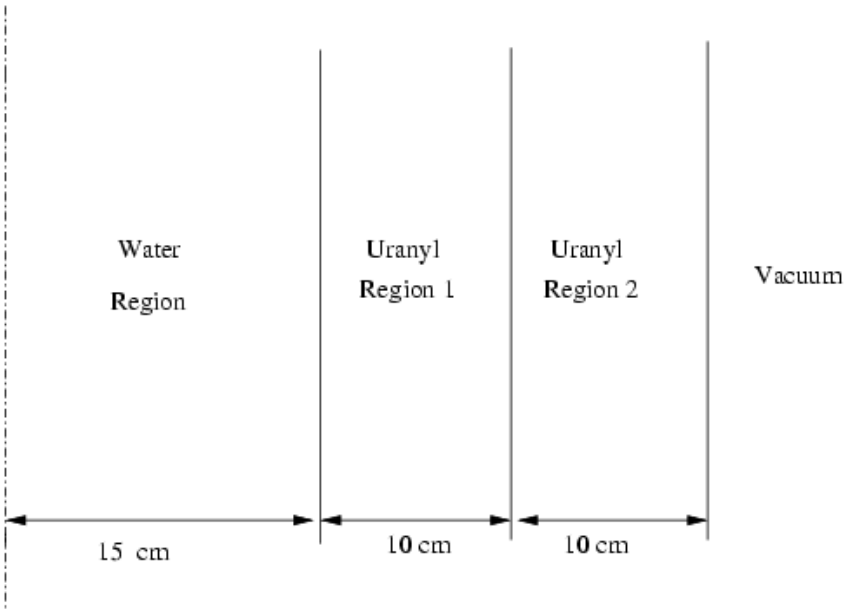

Figure 3.2: The model used for the cross section generation problem

According to make this analysis two group structure was used. In this structure thermal group is between 0 and 0.625 eV and fast group is between 0.625 eV and 20 MeV .

3.4 Results

Using above relation cross sections are calculated and given in Table 3.4. In this cal-

Table 3.1: Cross section value calculated from MCNP

	Water Region		Uranyl Region 1		Uranyl Region 2	
	fast	thermal	fast	thermal	fast	thermal
Σ_t	0.912137	2.072187	0.79533	2.1262	0.79334	1.89651
Σ_γ	5.749810^{-4}	0.019115	3.503310^{-3}	0.026963	3.4838710^{-3}	0.026743
ν	0	0	2.54	2.431	2.54	2.431
$\nu\Sigma_f$	0	0	0.0020192	0.0864	0.0019997	0.0856456

culation MCNP give effective multiplication factor (k_{eff}) 0.91842.

One of the main problems when using the diffusion equation to describe the

neutron balance in the system is how to define the homogenized diffusion coefficient. The primary role of the diffusion coefficient in the balance equation is to account for the anisotropy of the neutron transport and the node-to-node leakage. The diffusion coefficient could be defined in different ways, in order to preserve parameters of interest such as reaction rate, eigenvalue, surface leakage or node-averaged leakage.

Various methods have been used to define diffusion coefficients. Although most of the computations of diffusion coefficients are based on deterministic methods, there are also calculational methods based on Monte Carlo techniques. Three method for calculating the diffusion coefficient for a node,

- A Monte Carlo normalized diffusion method
- Use of the conventional definition $(3\Sigma_{tr})^{-1}$ for the diffusion coefficient, where the transport cross section Σ_{tr} is estimated from an MCNP simulation.
- An optimization method based on a genetic algoritms.

In the second method one must add a patch file to MCNP to find direction cosines. After this MCNP give Σ_{tr} and then diffusion coefficient can be found [8].

In our analysis we used first method. In this method, since cross sections are determined from the MCNP simulation of the lattice cell, a diffusion code (SLABNOD) was used iteratively. One group nodal diffusion calculation is performed at each iteration. And system multiplication constant and nodal fluxes are compared to MCNP results. According to make this iterations and comparison a shell script[9] was written. As a result, for uranyl region 1 diffusion coefficient found 3.005 cm and for uranyl region 2 diffusion coefficient found 2.09 cm. And nodal diffusion code calculate the effective multiplication factor 0.9184265 . According to show flux distribution on uranyl region, these regions are divided to ten node and slabnod code run again. These flux distribution given in Figure 3.3. In this figure first node is the node next to the water region in Figure 3.2 and 20. node is next to the vacuum region.

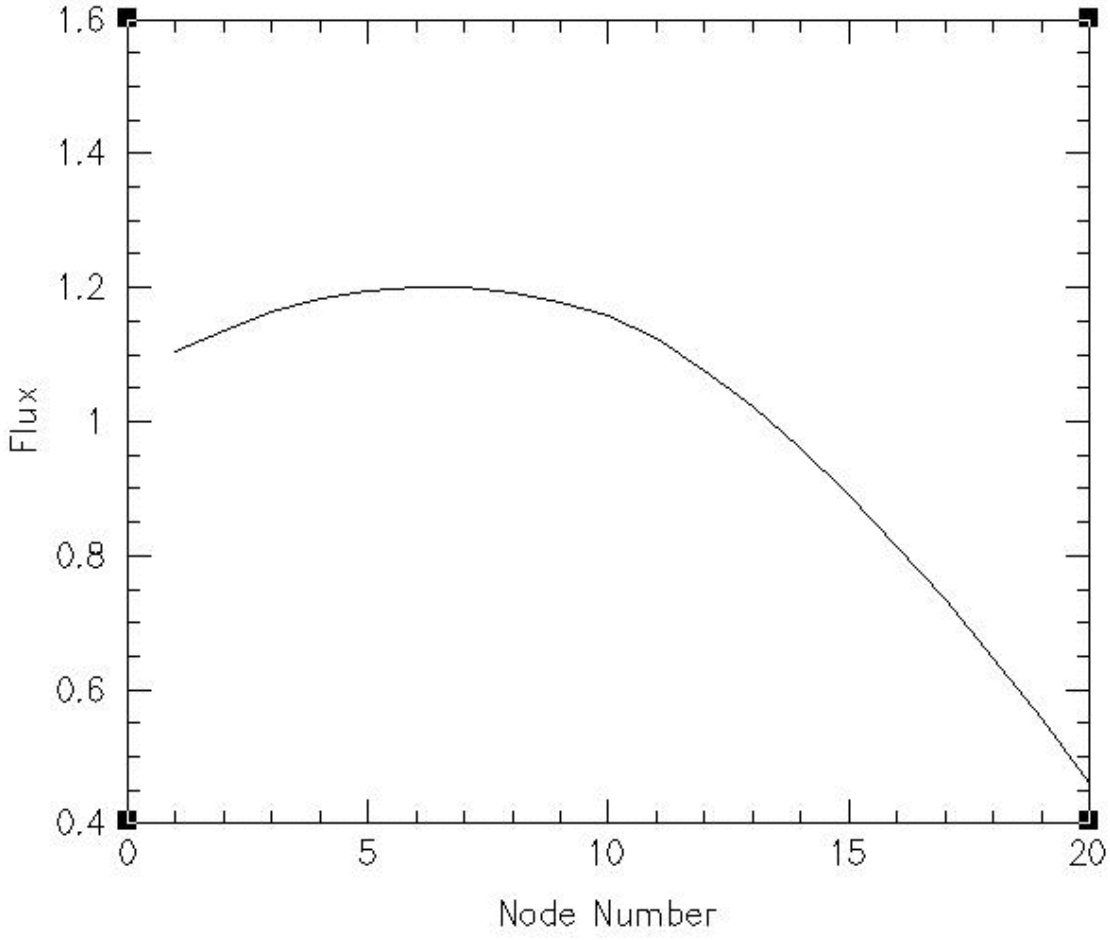

Figure 3.3: Flux distribution

Chapter 4

CONCLUSION

The main purpose of this study is to generate neutron cross sections and diffusion coefficients with using analytical nodal method and Monte Carlo technique.

In this thesis nodal diffusion theory is examined. Nodal diffusion equations derived for one dimensional slab geometry and cylinder geometry. The one and one half group neutron diffusion equations in one dimensional slab geometry was derived and a new nodal program was developed for the iterative solution of the resulting system of equations, which yield optimally fast solver. Results shows that the nodal methods and our algorithms working well.

For the cylindrical geometry nodal diffusion calculations we derive the equations and write a program utilizing these equations. And compare the results of this program with the analytical solution of the criticality condition for cylindrical geometry.

In the second part of the thesis we work on a methodology to generate multigroup cross sections. According to do this we use MCNP code which utilizes Monte Carlo methods. Monte Carlo methods of performing radiation transport calculations are heavily used in many different applications. Since the primary difficulty in using non continuous energy methods is the need for reliable multigroup cross sections that have been collapsed over an appropriate flux spectrum, we use Monte Carlo code MCNP to generate multigroup cross section. This will make it possible to take advan-

tage of the accuracy and ease of problem definition provided by MCNP while generating multi group cross sections for other applications such as nodal codes. The exact geometry of the problem can be modeled in MCNP so, spectrum obtained exactly and reliable results are obtained in the analysis.

Since diffusion coefficient does not obtained from MCNP directly we use different method. The slab geometry model is generated and used in MCNP to obtain cell neutron fluxes and reaction rates as well as multiplication factor. These results were used for obtaining homogenized node cross sections for each node. Nodal diffusion equations, which use modified one group theory, for slab and cylindrical geometry are derived and utilized in the analysis to determine diffusion coefficients of each node. Cell neutron fluxes, reaction rates, and multiplication factor were obtained for slab geometry model generating in MCNP. The fluxes and reaction rates were used in obtaining homogenized node cross sections for each node. These cross sections were used in nodal diffusion code (slabnod) and system multiplication constant and nodal fluxes are compared to MCNP results iteratively to determine diffusion coefficients of each node. Because of nodal codes are gives results quickly this methodology is fast and reliable since exact neutron spectrum was used.

Appendix A

CODE USED FOR THE ANALYSIS

For the neutronic analysis, MCNP code is used. In this appendix, brief descriptions of this code are introduced.

A.1 Monte Carlo Method

In the reactor analysis the problems which we have considered can also be solved with relative ease by another even more elemantary method: the method of the random walk. Rather than considering the neutrons which happen to be in a given volume element and which have definite velocities, this method follows the motion of the neutron through its successive collision, from the time it is emitted by the source to the time it is finally absorbed. Computation of the distribution function based on the random walk method may be adapted to the use of calculating machines even in non-homogeneous media. In this case, since successive events are regulated by accident or chance, the method is referred to as the *Monte Carlo* method [7]. Numerical methods that are known as Monte Carlo methods can be loosely described as statistical simulation methods, where statistical simulation is defined in quite general terms to be any method that utilizes sequences of random numbers to perform the simulation. Monte Carlo methods have been used for centuries, but only in the several decades has the tecnique gained the status of a full-pledged numerical method capable of addressing the most complex application.

A.2 MCNP

A General Monte Carlo N-Particle Transport (MCNP) code is being used widely for the analysis of experimental data, nuclear safeguards, radiation shielding, health physics problems, magnetic fusion neutronics, reactor designs, activation analysis and nuclear instrumentation design. By aid of improved options in MCNP and the development of fast running computing system, the use of MCNP is being extended to the reactor design studies and is used as a validation tool for the system of which the experimental data is not easily available.

The MCNP is a general-purpose Monte Carlo transport code [4] which solves for the generalized geometry, time-dependent and coupled neutron-photon problems with continuous or discrete energy structure. The interaction of neutron and/or photon is statistically calculated using the nuclear data libraries provided. For neutron calculations, all reactions are accounted for based on the reaction types specified in a particular cross-section libraries.

The name of method derives from the statistical sampling technique, which reminds of games of chance, where randomness would statistically become resolved in predictable probabilities, therefore Monte-Carlo. The Monte-Carlo method is a probabilistic method, which gives an estimate of the quantity of interest (reaction rates, scalar fluxes, etc.) calculating an average value for a finite number of histories or trajectories. The histories are randomly started and traced according to the elementary laws of physics, which determine the movement of neutrons and/or other particles.

Monte-Carlo methods are very different from deterministic transport methods. Deterministic methods, the most common of which, after diffusion theory, is the discrete ordinates method, solve the transport equation for the average particle behavior. By contrast, MCNP code does not solve an explicit equation, but rather simulates individual particles recording some aspects (tallies) of their average behavior. The average behavior of particles in the physical system is then inferred (using the central limit theorem) from the average behavior of the simulated particles.

The Monte-Carlo method can be used to simulate a statistical process (such

as the interaction of nuclear particles with materials) and is particularly useful for complex problems that cannot be modeled by computer codes that use deterministic methods. The individual probabilistic events that comprise a process are simulated sequentially. The probability distributions governing these events are statistically sampled to describe the total phenomenon, and the statistical sampling process is based on the selection of random numbers. In general, the number of trials necessary to adequately describe the phenomena is usually quite large. In particle transport, the Monte-Carlo technique is pre-eminently realistic (a theoretical experiment). It consists of actually following each of many particles from a source throughout its life to its end in some terminal category (absorption, escape, etc.). Probability distributions are randomly sampled using transport data to determine the outcome at each step of its life. Figure A.1 represents the random history of a neutron incident on a slab of material that can undergo fission. Numbers between 0 and 1 are selected randomly to determine what (if any) and where interaction takes place, based on the rules (physics) and probabilities (transport data) governing the processes and materials involved. In this particular example, a neutron collision occurs at event 1. The neutron is scattered in the direction shown, which is selected randomly from the physical scattering distribution. A photon is also produced and can be temporarily stored, or banked, for later analysis. At event 2, fission occurs, resulting in the termination of the incoming neutron and the birth of two outgoing neutrons and one photon. One neutron and the photon are banked for later analysis. The first fission neutron is captured at event 3 and terminated. The banked neutron is now retrieved and, by random sampling, leaks out of the slab at event 4. The fission-produced photon has a collision at event 5 and leaks out at event 6. The remaining photon generated at event 1 is now followed with a capture at event 7. Note that MCNP retrieves banked particles such that the last particle stored in the bank is the first particle taken out. This neutron history is now complete. As more and more such histories are followed, the neutron and photon distributions and the associated events become better known. The quantities of interest (whatever the user requests) are tallied, along with estimates of the statistical precision (uncertainty) of the results.

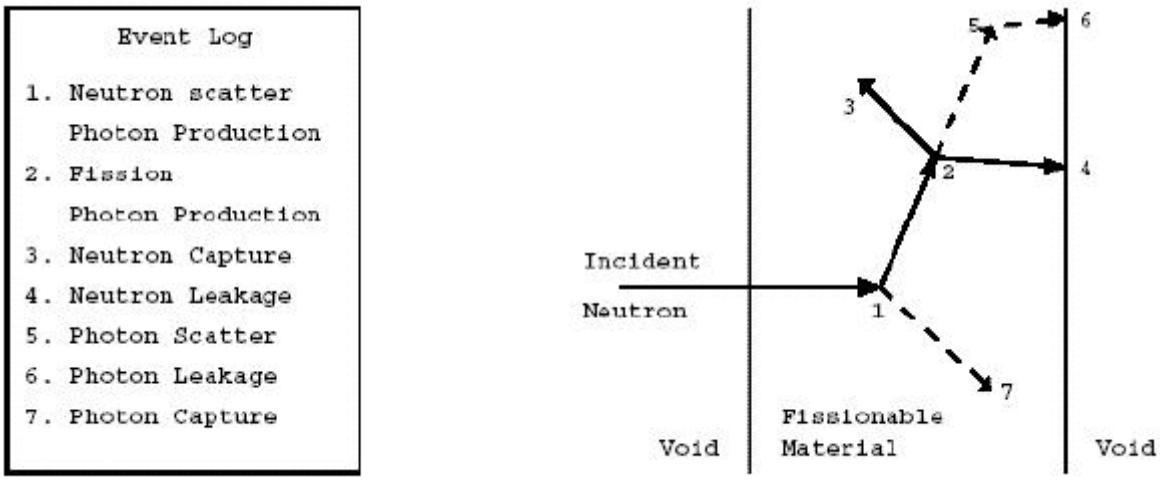

Figure A.1: The Monte Carlo method

Calculating k-effective consists of estimating the mean number of fission neutrons produced in one generation per fission neutron started. A generation is the life of a neutron from the origin in fission to the end by escape, parasitic capture, or absorption leading to fission. In MCNP, the computational equivalent of a fission generation is a k-effective cycle; i.e. a cycle is a computed estimate of an actual fission generation. Processes such as (n,2n) and (n,3n) are considered internal to a cycle and do not act as termination. Because fission neutrons are terminated in each cycle to provide the fission source for the next cycle, a single history can be viewed as continuing from cycle to cycle. The effect of the delayed neutrons is included using the total number of neutron generated per fission and their relative spectrum. It is imperative to emphasize that the result from a criticality calculation is a confidence interval for k-effective that is formed using the final estimated k-effective and the estimated standard deviation.

A properly formed confidence interval from a valid calculation should include the true answer. There will always be some probability that the true answer lies outside of a confidence interval.

On the other hand, MCNP has some limitations and inconvenient characteristics. The accuracy of MCNP results is limited by the statistical nature of the model.

Results can be obtained with any desired statistical certainty, provided that a sufficient number of histories is followed. The larger the number of histories, the better the results, however, also, the longer MCNP runs. MCNP has some variance reduction options for increasing the accuracy without increasing the number of histories.

MCNP can provide only the reaction rates and fluxes for each defined cell in the model. To obtain cross sections some post processing is required. Tallying is possible only over the defined cells and surfaces of the model. If the desired cells can not be modeled, the post processing of the data is required. MCNP does not provide group to group scattering reaction rates. Therefore it is impossible to obtain group to group scattering cross sections directly from MCNP results.

Appendix B

SLABNOD

A programming quote, ”The best performance improvement is the transition from the nonworking state to the working state”.

SLABNOD is written in FORTRAN-77 programming language. It consist of three subroutines (coefref, coefleft, and coefcenter). In these subroutines, according to nodal methods, fluxes are calculated. From these fluxes system reactivity found and then compared with previous iteration results. It continues until the fluxes and reactivities converged.

In this appendix the code and sample input given.

B.1 Slabnod Code

```
      program diff
      implicit real (a-h,o-z)
c     parameter(n=3)
c     nm is the total number of region with different neutronic parameters
c     nr is total number of region
c     ni is the maximum iteration number

      integer s1,s2,s9
      integer ITT(0:50),IAN(50)
      real FHI(0:50)
      real FHIOLD(0:50),rho(0:50),miga(0:50),difc(0:50),xl(0:50)

      open(1,file='1nosym1d.inp',form='formatted',status='old')
```

```
      open(20,file='1nosym1d.out',status='unknown')
c     read assembly types
      read(1,'(3i7)') nr,nm,ni
      write(20,'(3i7)') nr,nm,ni
      read(1,'(2i3)')(itt(i), i=1,nr)
         do i=1,nr
            read(1,*) xl(i)
            write(20,'(f8.3)') xl(i)
         enddo
         do i=1,nm
            read(1,*) rho(i),miga(i),difc(i)
            write(20,*) rho(i),miga(i),difc(i)
         enddo
c     alpha j=alpha*phi, albedo
         read(1,*) alpha
         read(1,*) alpha2
         write(*,*) alpha,alpha2
         write(20,*) alpha,alpha2
C     COMPUTE NUMBER OF ASSYS OF EACH TYPE A
C     XNA total # of assemblies
C     ian(i) total # of assemblies of  type i
      XNA=0
      DO i=1,nm
         IAN(I)=0
      enddo
      xlengtot=0.0
      DO I=1,nr
         K=ITT(I)
         IF (K.ne.0) then
         IAN(K)=IAN(K)+1
         endif
         xlengtot= xlengtot+xl(i)
       enddo
       write(20,*) 'xlengtot',xlengtot
       write(*,*) 'xlengtot',xlengtot
       DO I=1,NM
         XNA=XNA+IAN(I)
       enddo
       write(*,*)'total number of assemblies',xna
C     START OF MAIN ITERATION ROUTINE
      do i=1,nr
            fhiold(i)=0.
            fhi(i)=0.
      enddo
c     flux initialization
```

```
      do i=1,nr
            if(itt(i).ne.0)then
               fhi(i)=1./xl(i)
               fhiold(i)=1./xl(i)
            endif
         enddo
      FL=0
      FR=0
      it=0
 1256 fs=0.
      ft=0.
      ftt=0.

      do i=1,nr
            s9=itt(i)
            FT=FT+Fhi(I)*difc(s9)/((1.-rho(s9))*miga(s9))
            FTT=FTT+Fhi(I)*xl(i)
            Fs=Fs+Fhi(I)*rho(s9)*difc(s9)/((1.-rho(s9))*miga(s9))
      enddo
c     we need to calculate the leakage reactivities for the
c     peripheral assemblies
          fl=0.0001
          rhos=(fs-fl)/ft
c     rhos is the reactivity of the system as an initial guess
       itt(0)=1
       xl(0)=1
 2100 continue

      ft=0.
      fl=0.
      fr=0.
      ftt=0.
      DO 2630 I=1,nr
c            IF (I.lE.1) goto 1111
c left node properties
c aa is used for the left node aa=(kinf/k-1)/migarea
c node size need to be used xl(i-1)
            S1=ITT(I-1)
            aa=(rho(s1)-rhos)/(miga(s1)*(1.-rho(s1)))
            xa=xl(i-1)
c right node properties
c bb is used for the right node bb=(kinf/k-1)/miga
c node size need to be used xl(i+1)

            if (i .ge. nr) goto 1111
```

```
            S2=ITT(I+1)
            bb=(rho(s2)-rhos)/(miga(s2)*(1.-rho(s2)))
            xb=xl(i+1)
c centeral node properties
c c is used for the right node cc=(kinf/k-1)/miga
c node size need to be used xl(i)
 1111       continue
            S9=ITT(I)
            cc=(rho(s9)-rhos)/(miga(s9)*(1.-rho(s9)))
            xc=xl(i)
            d1=difc(s9)/difc(s1)
            d2=difc(s9)/difc(s2)
            if (i .eq. nr) then
               doalphah=difc(s9)/alpha
       call coefref(aa,cc,d1,xa,xc,
     #doalphah,alpha,xleft,center,rq1,rq2)
       Fhi(I)=xleft*fhi(i-1)/center
       write(20,*) xleft,xcenter,'xleft,center'
      else if (i.eq.1) then
              doalpha=difc(s9)/alpha2
          call coefleft(bb,cc,d2,xb,xc,doalpha,alpha2,xright,center,rq1,rq2)
          Fhi(I)=xright*fhi(i+1)/center
      else
       call coefcenter(aa,bb,cc,d1,d2,xa,xb,xc,xfl,xfr,xfc)
       Fhi(I)=(xfl*fhi(i-1)+xfr*fhi(i+1))/xfc
       write(20,*) xfl,xfr,xfc,aa,bb,cc,d1,d2,x, 'xfl,xfr,xfc,aa,bb,cc'
      endif
c  *************** f values updated
            IF (I.EQ.nr) then
c leakage reactivity calculation
            fl=fl+Fhi(I)*rQ1*difc(s9)
            endif
            if (i.eq.1) then
             fl=fl+Fhi(I)*rQ1*difc(s9)
            endif

 2540       FT=FT+Fhi(I)*difc(s9)/((1.-rho(s9))*miga(s9))
            FTT=FTT+Fhi(I)*xl(i)
            FR=FR+Fhi(I)*rho(s9)*difc(s9)/((1.-rho(s9))*miga(s9))

 2630 CONTINUE
      RC=FR/FT
      RL=FL/FT
      write(*,*) rc,rl,ft,'rc,rl,ft'
C     CALCULATE  REACTIVITY AS DIFFERENCE OF CORE AND LEAKAGE
```

```
      rhosold=rhos
      rhos=RC-RL
      DO I=1,nr
C     NORMALIZE POWERS
c          Write(20,*) i,fhi(i),ftt,xna
          Fhi(I)=(Fhi(I)/FTT)*xlengtot

            Write(20,*) i,fhi(i),ftt,xna
         enddo
         ftt=0.
         IT=IT+1
      IF (IT.GT.NI) then
         goto 2850
      else
         if(it.gt.3) then

            do i=1,nr
               arg=abs(fhi(i)-fhiold(i))
                  if(arg.gt.0.000001)goto 113
               enddo
             arg=abs(rhos-rhosold)
            if(arg.gt.0.0000001)goto 113
            goto 2850
         end if
      end if
 113  continue
      do i=1,nr
            k=itt(i)
            if(k.eq.0) goto 2855
            fhiold(i)=fhi(i)
 2855    enddo
 2840 GOTO 2100

 2850 continue
      do i=1,nr
         write(20,'(f9.4,i3)') fhi(i),i
      enddo
      write(20,*)'iteration number,final rhos'
      write(20,*) it,rhos
 99   FORMAT(40X,F7.4,2x,i3)
      stop
      end

      Subroutine coefcenter(aa,bb,cc,d1,d2,xa,xb,xc,fl,fr,fc)
c     this program is used to calculate
```

```
c     coefficients of computational molecule to
c     perform nodal diffusion calculations
c     fc is the value of the coefficient for center
c     fl is the value of the coefficient for the left node
c     fr is the value of the coefficient for the rigt node
c     ft is the value of the coefficient for the top node
c     fb is the value of the coefficient for the bottom node

c     input parameters used for evaluations
c     d1 and d2 is ratios of diffusion coefficients , h mesh size
c     b=center, a=left and c=right, alpha=albedo

c     for a given set of a, b, and c we want to determine
c     which function is used in our calculations
c     left (a*x/sinh(a*x) if a gt 0 else (a*x/sin(a*x))
      if (aa .gt. 0) then
         a=sqrt(aa)
         aa1l=(a/sin(a*xa))
        else
         a=sqrt(-aa)
         aa1l= (a/sinh(a*xa))
      endif

      if (cc .gt. 0) then
         c=sqrt(cc)
               cc1=(sin(c*xc/2.)/(c*cos(c*xc/2.)))
            else
         c=sqrt(-cc)
               cc1=(sinh(c*xc/2.)/(c*cosh(c*xc/2.)))
      endif

      if(bb .gt. 0) then
       b=sqrt(bb)
         bb1c=sin(b*xb/2.)/(b*cos(b*xb/2.))
       else
        b=sqrt(-bb)
          bb1c=sinh(b*xb/2.)/(b*cosh(b*xb/2.))
          endif

c      for left

       fl=aa1l*(cc1+d2*bb1c)

c     left (b*x/sinh(b*x) if a gt 0 else (b*x/sin(b*x))
```

```
        if (bb .gt. 0) then
          b=sqrt(bb)
          bb1r=(b/sin(b*xb))
         else
          b=sqrt(-bb)
          bb1r= (b/sinh(b*xb))
         endif

       if(aa .gt. 0) then
        a=sqrt(aa)
          aa1c=sin(a*xa/2.)/(a*cos(a*xa/2.))
        else
         a=sqrt(-aa)
           aa1c=sinh(a*xa/2.)/(a*cosh(a*xa/2.))
           endif

c      for right

        fr=bb1r*(cc1+d1*aa1c)

       if (cc .gt. 0) then
          c=sqrt(cc)
                cc1c=-(cos(c*xc/2.)/(c*sin(c*xc/2.)))
             else
          c=sqrt(-cc)

                cc1c=(cosh(c*xc/2.)/(c*sinh(c*xc/2.)))
        endif

        ccc=-cc
        fc=ccc*((cc1c+d1*aa1c)*(cc1+d2*bb1c)+
      #(cc1c+d2*bb1c)*(cc1+d1*aa1c))/2.
        return
        end


        subroutine coefref(aa,cc,d1,xa,xc,
      #doalphah,alpha,flref,fcref,rq1,rq2)

c      this program is used to calculate
c      coefficients of computational molecule to
c      perform nodal diffusion calculations

c       input parameters used for evaluations
c      d1 and d2 is ratios of diffusion coefficients , x mesh size
```

```
c     b=center, a=left and c=right, alpha=albedo
c     d1=dij/di-1,j and d2=dij/di+1j

c     for a given set of a, b, and c we want to determine
c     which function is used in our calculations
c     left (a*x/sinh(a*x) if a gt 0 else (a*x/sin(a*x))
      if (aa .gt. 0) then
         a=sqrt(aa)
         aa1l=(a/sin(a*xa))
        else
         a=sqrt(-aa)
         aa1l= (a/sinh(a*xa))
      endif

      if (cc .gt. 0) then
         c=sqrt(cc)
               cc1=(sin(c*xc/2.)/(c*cos(c*xc/2.)))
            else
         c=sqrt(-cc)
               cc1=(sinh(c*xc/2.)/(c*cosh(c*xc/2.)))
      endif

      if (aa .gt. 0) then
         a=sqrt(aa)
               aa1=(sin(a*xa/2.)/(a*cos(a*xa/2.)))
            else
         a=sqrt(-aa)
               aa1=(sinh(a*xa/2.)/(a*cosh(a*xa/2.)))
      endif

      if (cc .gt. 0) then
         c=sqrt(cc)
               cc1c=-(cos(c*xc/2.)/(c*sin(c*xc/2.)))
            else
         c=sqrt(-cc)

               cc1c=(cosh(c*xc/2.)/(c*sinh(c*xc/2.)))
      endif

c     peripheral nodes
c     flref,fcref
      flref=aa1l
      ccc=-cc
      fcref=ccc*((cc1c+d1*aa1)+(cc1+d1*aa1)*
     #(cc1c+doalphah)/(cc1+doalphah))/2
```

```
      rq1=-ccc*(1.-(cc1c+doalphah)/(cc1+doalphah))/2.

      rq2=ccc*(cc1c-cc1)/(doalphah+cc1)/2
      rq3=ccc*(cc1c*cc1c*ccc-1.)/
     #(1.+doalphah*ccc*cc1c)/2
      return
      end

      subroutine coefleft(aa,cc,d1,xa,xc,
     #doalphah,alpha,flref,fcref,rq1,rq2)

c     this program is used to calculate
c     coefficients of computational molecule to
c     perform nodal diffusion calculations

c      input parameters used for evaluations
c     d1 and d2 is ratios of diffusion coefficients , x mesh size
c     b=center, a=left and c=right, alpha=albedo
c     d1=dij/di-1,j and d2=dij/di+1j

c     for a given set of a, b, and c we want to determine
c     which function is used in our calculations
c     left (a*x/sinh(a*x) if a gt 0 else (a*x/sin(a*x))
      if (aa .gt. 0) then
         a=sqrt(aa)
         aa1l=(a/sin(a*xa))
        else
         a=sqrt(-aa)
         aa1l= (a/sinh(a*xa))
      endif

      if (cc .gt. 0) then
         c=sqrt(cc)
               cc1=(sin(c*xc/2.)/(c*cos(c*xc/2.)))
            else
         c=sqrt(-cc)
               cc1=(sinh(c*xc/2.)/(c*cosh(c*xc/2.)))
      endif

      if (aa .gt. 0) then
         a=sqrt(aa)
               aa1=(sin(a*xa/2.)/(a*cos(a*xa/2.)))
            else
         a=sqrt(-aa)
               aa1=(sinh(a*xa/2.)/(a*cosh(a*xa/2.)))
```

```
      endif

      if (cc .gt. 0) then
         c=sqrt(cc)
               cc1c=-(cos(c*xc/2.)/(c*sin(c*xc/2.)))
            else
         c=sqrt(-cc)

               cc1c=(cosh(c*xc/2.)/(c*sinh(c*xc/2.)))
      endif

c     peripheral nodes
c     flref,fcref
      flref=aa1l
      ccc=-cc
      fcref=ccc*((cc1c+d1*aa1)+(cc1+d1*aa1)*
     #(cc1c+doalphah)/(cc1+doalphah))/2
      rq1=-ccc*(1.-(cc1c+doalphah)/(cc1+doalphah))/2.

      rq2=ccc*(cc1c-cc1)/(doalphah+cc1)/2
      rq3=ccc*(cc1c*cc1c*ccc-1.)/
     #(1.+doalphah*ccc*cc1c)/2

      return
      end
```

B.2 Input for Slabnod

```
     4       3  9999
 3 1
 2 2
 1.158
 3.231
 3.231
 3.231
0.242038 52.101 1.24
0.317308 42.958 1.22
-0.3121 702.439 1.44
```

```
0.0000001
0.5
```

The output of this gives the system reactivity as -0.144509763, that is, $k_{eff} = 0.87373654$. And fluxes are given in the table below as expected since there is no leakage in the right boundary and left boundary is vacuum.

Table B.1: Fluxes of the sample run results

Node	1	2	3	4
Node Flux	0.1613	0.7589	1.1587	1.3830

Appendix C

CYLNOD

```
      program cylnod
      implicit real (a-h,o-z)
c     parameter(n=3)
c     nm is the total number of region with different neutronic parameters
c     nr is total number of region
c     max number of region is 10
      integer s1,s2,s9
      integer ITT(0:44),IAN(22)
      real FHI(0:44),FHIOLD(0:44),qrady(0:44),alp(0:44)
      real rho(0:22),miga(0:22),difc(0:22),xl(0:44),rad(0:44)
      real pi
      PI=3.141592654
      open(1,file='cylnod1d.inp',form='formatted',status='old')
      open(20,file='cyl1d1d.out',status='unknown')
      read(1,'(3i5)') nr,nm,ni
      write(20,'(3i5)') nr,nm,ni
      read(1,'(2i3)')(itt(i), i=1,nr)
         do i=1,nr
            read(1,*) xl(i)
            write(20,'(f8.3)') xl(i)
         enddo
         do i=1,nm
            read(1,*) rho(i),miga(i),difc(i)
            write(20,*) rho(i),miga(i),difc(i)
         enddo
c     alpha j=alpha*phi, albedo
         read(1,*) alpha
         write(20,*) alpha
      XNA=0
      DO i=1,nm
```

```
         IAN(I)=0
      enddo
      qrady(0)=0.
      qrady(1)=xl(1)
      DO i=2,nr
       qrady(i)=qrady(i-1)+xl(i)
      enddo
      DO I=1,nr
         K=ITT(I)
         IF (K.ne.0) then
         IAN(K)=IAN(K)+1
         endif
       enddo
       DO I=1,NM
         XNA=XNA+IAN(I)
       enddo
C     START OF MAIN ITERATION ROUTINE
      do i=1,nr
            fhiold(i)=0.
            fhi(i)=0.
      enddo
c     flux initialization
      do i=1,nr
               fhi(i)=qrady(i)*besj1(qrady(i)*2.405/qrady(nr))-
     &           qrady(i-1)*besj1(qrady(i-1)*2.405/qrady(nr))
               fhi(i)=1
               fhiold(i)=1
      enddo
      FL=0
      FR=0
      it=0
 1256 fs=0.
      ft=0.
      ftt=0.
      do i=1,nr
            s9=itt(i)
            FT=FT+Fhi(I)*difc(s9)/((1.-rho(s9))*miga(s9))
            FTT=FTT+Fhi(I)
            Fs=Fs+Fhi(I)*rho(s9)*difc(s9)/((1.-rho(s9))*miga(s9))
      enddo
         rhos=fs/ft-0.000005
 2100 continue
      ft=0.
      fl=0.
      fr=0.
```

```
      ftt=0.
      DO 2630 I=1,nr-1
            S9=ITT(I)
            cc=(rho(s9)-rhos)/(miga(s9)*(1.-rho(s9)))
           xa=qrady(i-1)
           xb=qrady(i)
       if(i.eq.1) then
        call coeffirst(cc,difc(s9),xb,alp(i),fhi(i),fii)
       else
       call coefcenter(cc,difc(s9),xa,xb,fii,alp(i-1),fhi(i),fi2,alp(i))
        fii=fi2
       endif
            FTT=FTT+Fhi(I)
            FT=FT+Fhi(I)*difc(s9)/((1.-rho(s9))*miga(s9))
            FR=FR+Fhi(I)*rho(s9)*difc(s9)/((1.-rho(s9))*miga(s9))
 2630 CONTINUE
        i=nr
         S9=ITT(I)
           cc=(rho(s9)-rhos)/(miga(s9)*(1.-rho(s9)))
           xa=qrady(i-1)
           xb=qrady(i)
         alpmt1=alpmt1(i-1)

       call coeflast(cc,difc(s9),xa,xb,fii,alp(i-1),fhi(i),fi2,alp(i))
       alpmt2=alp(i-1)
            FTT=FTT+Fhi(I)
            FT=FT+Fhi(I)*difc(s9)/((1.-rho(s9))*miga(s9))
            FR=FR+Fhi(I)*rho(s9)*difc(s9)/((1.-rho(s9))*miga(s9))
            FL=pi*xb*fi2

            fhinorm=1.0
       DO I=1,nr
C     NORMALIZE POWERS
          Fhi(I)=Fhi(I)/fhinorm
       enddo

      RC=FR/FT
      RL=FL/FT

      rhos=rhos*0.95+0.05*(RC-RL)

         ftt=0.
         IT=IT+1
      IF (IT.GT.NI) then
         goto 2850
```

```
      else
         if(it.gt.2) then

            do i=1,nr
               arg=abs((fhi(i)-fhiold(i))/fhi(i))
                  if(arg.gt.0.0001)goto 113
               enddo
             arg=abs(rhos-rhosold)
            if(arg.gt.0.000001)goto 113
            goto 2850
         end if
      end if
 113  continue
      do i=1,nr
            k=itt(i)
            if(k.eq.0) goto 2855
            fhiold(i)=fhi(i)
              rhosold=rhos
 2855    enddo
 2840 GOTO 2100
 2850 continue
        write(*,*) 'fhi ,i ,it ,rho'
        write(20,*) 'fhi ,i ,it ,rho'
        do i=1,nr
        write(*,*) fhi(i)/pi/(qrady(i)**2-qrady(i-1)**2),i,it,rhos
        write(20,*) fhi(i)/pi/(qrady(i)**2-qrady(i-1)**2),i,it,rhos
        enddo
 99   FORMAT(40X,F7.4,2x,i3)
      stop
      end

      Subroutine coefcenter(cc,d1,xa,xb,fii,alpi,flux,fi2,alp2)
      real pi
      PI=4.*atan(1.0)

        if(cc.gt.0) then
        c=sqrt(cc)
        eps1=besj0(c*xa)-2.*d1*c*besj1(c*xa)
        eps2=besy0(c*xa)-2.*d1*c*besy1(c*xa)
        eps3=besj0(c*xa)+2.*d1*c*besj1(c*xa)
        eps4=besy0(c*xa)+2.*d1*c*besy1(c*xa)

        gi=fii/(besy0(c*xa)+besj0(c*xa)*(alpi*eps4-eps2)/
&       (eps1-alpi*eps3))
        ai=gi*(alpi*eps4-eps2)/(eps1-alpi*eps3)
```

```
        flux=2.*pi/c*(ai*(xb*besj1(c*xb)-xa*(besj1(c*xa)))+
&       gi*(xb*besy1(c*xb)-xa*besy1(c*xa)))

        fi2=ai*besj0(c*xb)+gi*besy0(c*xb)
        fi3=ai*besj0(c*xa)+gi*besy0(c*xa)

        eps11=besj0(c*xb)-2.*d1*c*besj1(c*xb)
        eps12=besy0(c*xb)-2.*d1*c*besy1(c*xb)
        eps13=besj0(c*xb)+2.*d1*c*besj1(c*xb)
        eps14=besy0(c*xb)+2.*d1*c*besy1(c*xb)

        else
        c=sqrt(-cc)
        eps1=bessi0(c*xa)+2.*d1*c*bessi1(c*xa)
        eps2=bessk0(c*xa)-2.*d1*c*bessk1(c*xa)
        eps3=bessi0(c*xa)-2.*d1*c*bessi1(c*xa)
        eps4=bessk0(c*xa)+2.*d1*c*bessk1(c*xa)

        gi=fii/(bessk0(c*xa)+bessi0(c*xa)*(alpi*eps4-eps2)/
&       (eps1-alpi*eps3))
        ai=gi*(alpi*eps4-eps2)/(eps1-alpi*eps3)

        flux=2.*pi/c*(ai*(xb*bessi1(c*xb)-xa*(bessi1(c*xa)))-
&       gi*(xb*bessk1(c*xb)-xa*bessk1(c*xa)))

        fi2=ai*bessi0(c*xb)+gi*bessk0(c*xb)

        eps11=bessi0(c*xb)+2.*d1*c*bessi1(c*xb)
        eps12=bessk0(c*xb)-2.*d1*c*bessk1(c*xb)
        eps13=bessi0(c*xb)-2.*d1*c*bessi1(c*xb)
        eps14=bessk0(c*xb)+2.*d1*c*bessk1(c*xb)

        endif
        alp2=(ai*eps11+gi*eps12)/(ai*eps13+gi*eps14)

      return
      end

      subroutine coeffirst(cc,d1,xb,alp1,fc,fii)
      real pi
      PI=4.*atan(1.0)
      if (cc .gt. 0) then
         c=sqrt(cc)
```

```
         alp1=(besj0(c*xb)-2.*d1*c*besj1(c*xb))/(besj0(c*xb)+
&        2.*d1*c*besj1(c*xb))
         fii=c*xb*besj0(c*xb)/(2.*besj1(c*xb))
      else
         c=sqrt(-cc)
         alp1=(bessi0(c*xb)+2.*d1*c*bessi1(c*xb))/(bessi0(c*xb)-
&        2.*d1*c*bessi1(c*xb))

         fii=c*xb*bessi0(c*xb)/(2.*bessi1(c*xb))
      endif
        fc=pi*xb*xb
      return
      end

      Subroutine coeflast(cc,d1,xa,xb,fii,alpi,flux,fi2,alp2)
      real pi
      PI=4.*atan(1.0)
      alp2=0.0

        if(cc.gt.0) then
        c=sqrt(cc)
        eps1=besj0(c*xb)-2.*d1*c*besj1(c*xb)
        eps2=besy0(c*xb)-2.*d1*c*besy1(c*xb)
        eps3=besj0(c*xb)+2.*d1*c*besj1(c*xb)
        eps4=besy0(c*xb)+2.*d1*c*besy1(c*xb)

        gi=fii/(besy0(c*xa)+besj0(c*xa)*(alp2*eps4-eps2)/
&       (eps1-alp2*eps3))
        ai=gi*(alp2*eps4-eps2)/(eps1-alp2*eps3)

        flux=2.*pi/c*(ai*(xb*besj1(c*xb)-xa*(besj1(c*xa)))+
&       gi*(xb*besy1(c*xb)-xa*besy1(c*xa)))

        fi2=ai*besj0(c*xb)+gi*besy0(c*xb)
        fi3=ai*besj0(c*xa)+gi*besy0(c*xa)

        eps11=besj0(c*xa)-2.*d1*c*besj1(c*xa)
        eps12=besy0(c*xa)-2.*d1*c*besy1(c*xa)
        eps13=besj0(c*xa)+2.*d1*c*besj1(c*xa)
        eps14=besy0(c*xa)+2.*d1*c*besy1(c*xa)

        else
        c=sqrt(-cc)
        eps1=bessi0(c*xb)+2.*d1*c*bessi1(c*xb)
        eps2=bessk0(c*xb)-2.*d1*c*bessk1(c*xb)
```

```
        eps3=bessi0(c*xb)-2.*d1*c*bessi1(c*xb)
        eps4=bessk0(c*xb)+2.*d1*c*bessk1(c*xb)

        gi=fii/(bessk0(c*xa)+bessi0(c*xa)*(alp2*eps4-eps2)/
&       (eps1-alp2*eps3))
        ai=gi*(alp2*eps4-eps2)/(eps1-alp2*eps3)

        flux=2.*pi/c*(ai*(xb*bessi1(c*xb)-xa*(bessi1(c*xa)))-
&       gi*(xb*bessk1(c*xb)-xa*bessk1(c*xa)))

        fi2=ai*bessi0(c*xb)+gi*bessk0(c*xb)

        eps11=bessi0(c*xa)+2.*d1*c*bessi1(c*xa)
        eps12=bessk0(c*xa)-2.*d1*c*bessk1(c*xa)
        eps13=bessi0(c*xa)-2.*d1*c*bessi1(c*xa)
        eps14=bessk0(c*xa)+2.*d1*c*bessk1(c*xb)

        endif
        alpi=(ai*eps11+gi*eps12)/(ai*eps13+gi*eps14)
      return
      end

       FUNCTION bessi0(x)
       REAL bessi0,x
c      Returns the modified Bessel function I0(x) for any real x.
       REAL ax
       DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,
&      q8,q9,y
c      Accumulate polynomials in double precision.
       SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
       DATA p1,p2,p3,p4,p5,p6,p7/1.0d0,3.5156229d0,3.0899424d0,
&      1.2067492d0,0.2659732d0, 0.360768d-1, 0.45813d-2/
       DATA q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,0.1328592d-1,
&      0.225319d-2,-0.157565d-2,0.916281d-2,-0.2057706d-1,
&      0.2635537d-1,-0.1647633d-1,0.392377d-2/
       if (abs(x).lt.3.75) then
       y=(x/3.75)**2
       bessi0=p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7)))))
       else
       ax=abs(x)
       y=3.75/ax
       bessi0=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4
&      +y*(q5+y*(q6+y*(q7+y*(q8+y*q9))))))))
        endif
        return
```

```
        END
        FUNCTION bessk0(x)
        REAL bessk0,x
C USES bessi0 Returns the modified Bessel function K0(x) for positive real x.
        REAL bessi0
        DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,
&       q2,q3,q4,q5,q6,q7,y
c    Accumulate polynomials in double precision.
        SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
       DATA p1,p2,p3,p4,p5,p6,p7/-0.57721566d0,0.42278420d0,
&      0.23069756d0,0.3488590d-1,0.262698d-2,0.10750d-3,0.74d-5/
       DATA q1,q2,q3,q4,q5,q6,q7/1.25331414d0,-0.7832358d-1,
&      0.2189568d-1,-0.1062446d-1,0.587872d-2,-0.251540d-2,0.53208d-3/
       if (x.le.2.0) then
c       Polynomial fit.
       y=x*x/4.0
       bessk0=(-log(x/2.0)*bessi0(x))+(p1+y*(p2+y*(p3+
&      y*(p4+y*(p5+y*(p6+y*p7))))))
       else
       y=(2.0/x)
       bessk0=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+
&       y*(q4+y*(q5+y*(q6+y*q7))))))
       endif
       return
       END

       FUNCTION bessi1(x)
       REAL bessi1,x
c       Returns the modified Bessel function I1(x) for any real x.
       REAL ax
       DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,
&      q8,q9,y
c   Accumulate polynomials in double precision.
       SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
       DATA p1,p2,p3,p4,p5,p6,p7/0.5d0,0.87890594d0,0.51498869d0,
&       0.15084934d0,0.2658733d-1,0.301532d-2,0.32411d-3/
       DATA q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,-0.3988024d-1,
&      -0.362018d-2,0.163801d-2,-0.1031555d-1,0.2282967d-1,
&      -0.2895312d-1,0.1787654d-1,-0.420059d-2/
       if (abs(x).lt.3.75) then
c       Polynomial fit.
       y=(x/3.75)**2
       bessi1=x*(p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
       else
       ax=abs(x)
```

```
      y=3.75/ax
      bessi1=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+
&     y*(q5+y*(q6+y*(q7+y*(q8+y*q9)))))))
      if(x.lt.0.)bessi1=-bessi1
      endif
      return
      END
      FUNCTION bessk1(x)
      REAL bessk1,x
C  USES bessi1 Returns the modified Bessel function K1(x) for positive real x.
      REAL bessi1
      DOUBLE PRECISION p1,p2,p3,p4,p5,p6,p7,q1,
&     q2,q3,q4,q5,q6,q7,y
c  Accumulate polynomials in double precision.
      SAVE p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      DATA p1,p2,p3,p4,p5,p6,p7/1.0d0,0.15443144d0,-0.67278579d0,
&     -0.18156897d0,-0.1919402d-1,-0.110404d-2,-0.4686d-4/
      DATA q1,q2,q3,q4,q5,q6,q7/1.25331414d0,0.23498619d0,
&     -0.3655620d-1,0.1504268d-1,-0.780353d-2,0.325614d-2,-0.68245d-3/
      if (x.le.2.0) then
      y=x*x/4.0
      bessk1=(log(x/2.0)*bessi1(x))+(1.0/x)*(p1+y*(p2+
&     y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
      else
      y=2.0/x
      bessk1=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+
&     y*(q4+y*(q5+y*(q6+y*q7))))))
      endif
      return
      END
```

Appendix D

MCNP INPUT FOR THREE REGION SYSTEM

```
 Cross section generation
1  2  9.9987e-2 1 -2 imp:n=1
2  1  9.9487e-2 2 -3 imp:n=1
3  1  9.9487e-2 3 -4 imp:n=1
4  0 #1 #2 #3 imp:n=0

*1 px 0
2 px 15
3 px 25
4 px 35

mode n
sdef x=d1
si1 0 35
sp1 0 1
kcode 2000 1.2 50 550
E0 6.25e-07 20
```

```
F4:n 2 3
SD4 1 1
Fm4 (7.64696e-5 35 -1) (7.64696e-5 35 -2) &
 (7.64696e-5 35 -6)  (7.64696e-5 35 -7) &
 (6.79526e-4 38 -1) (6.79526e-4 38 -2) &
 (6.79526e-4 38 -6) (6.79526e-4 38 -7) &
 (5.90425e-2 4 -1)  (5.90425e-2 4 -2) &
 (3.706467e-2 3 -1)  (3.706467e-2 3 -2) &
 (2.111e-3 5 -1) (2.111e-3 5 -2)
F14:n 1 2 3
SD14 1 1 1
F24:n 1
SD24 1
Fm24 (3.332466e-2 3 -1) (3.332466e-2 3 -2) &
 (6.664933e-2 4 -1)  (6.664933e-2 4 -2)
F34:n 1
SD34 1
F1:n 1 2 3 4
c1 0 1
M1 8016 3.72588e-2 92235 7.6864e-5 92238 6.8303e-4 &
 7014  2.122e-3 1001  5.9347e-2
M2 1001 6.6658e-2 8016  3.3329e-2
M3   8016 1
M38  92238 1
M4  1001 1
M5 7014 1
```

Bibliography

[1] Henry, A.F., 1980, Nuclear Reactor Analysis, The MIT Press, Cambridge, Massachusetts, 547p.

[2] Duderstadt, J., J., Hamilton, L., J., 1976, Nuclear Reactor Analysis, John Wiley&Sons, New York, 650p.

[3] Rahnema, F., McKinley, M., S., 2002,High-Order Cross Section Homogenization Method, Annals of Nuclear Energy, Volume 29, pp 875-899

[4] Briesmeister, J., F., Ed., 2000, MCNP Manual Version 4C, Los Alamos National Laboratory.

[5] Fowler, T., B., Vondy, D., R., and Cunningham, G., W., 1971, Nuclear Reactor Core Analysis Code: CITATION, ORNL-TM-2496, Rev.2, Oak Ridge National Laboratory.

[6] Yeong-Il Kim, Young-Jin Kim, Sang-Ji Kim, Taek-Kyum Kim, 1998, A Semi-analytic Multigroup Nodal Method, Annals of Nuclear Energy, Volume 26, pp 699-708

[7] Weinberg, A., M., Wigner, E., P., 1958, The Physical Theory of Neutron Chain Reactors, The University of Chicago Press, Chicago, Illinois, 801p.

[8] Ilas, G., Rahnema, F., 2002, A Monte Carlo Based Nodal Diffusion Model For Criticality Analysis of Spent Fuel Storage Lattices, Annals of Nuclear Energy, Volume 30, pp 1089-1108.

[9] Tansley, D., 2000, LINUX & UNIX Shell Programming, Addison-Wesley Professional, U.K., 528p.

[10] Monro, D., M., 1982, Fortran 77, Butler & Tanner Ltd., Great Britain, 360p.

[11] Stacey, W., M., 2001, Nuclear Reactor Physics, John Wiley&Sons, New York, 707p.

RESUME

Name Surname : Metin Köksal

Birth Place : Gölcük-Kocaeli

Birth Date : 30.06.1979

Marital Status : Single

Education and Academic Positions:

High School, Adana Fen Lisesi, 1993-1995

Tinaztepe Lisesi, 1995-1996

Undergraduate, Hacettepe Üniversitesi, Nükleer Enerji Mühendisliği, 1996-2003

Language : English

Experience :

2004- , Research Assistant at Hacettepe Üniversitesi, Nükleer Enerji Mühendisliği