A one-dimensional multi-group collision probability code for neutron transport analysis and criticality calculations

Student Name	**:**	**Sebenele Mugu Mtsetfwa**
Student Number	**:**	**22047360**
Supervisor	**:**	**Dr. Oscar Zamonsky**
Date	**:**	**23rd May 2012**

Mini-dissertation submitted in partial fulfilment of the requirements for the degree of Master of Science in Nuclear Engineering at the Potchefstroom Campus of the North-West University

ABSTRACT

This work develops a one dimensional, slab geometry, multigroup collision probability code named Oklo which solves both criticality calculations and fixed source problems. The code uses the classical collision probabilities approach where the first flight collision probabilities are calculated analytically for void, reflected and periodic boundary conditions.

The code has been verified against analytical criticality benchmark test sets from Los Alamos National Laboratory, which have been used to verify MCNP amongst other codes. The results from the code show a good agreement with the benchmark test sets for the critical systems presented in this report.

The results from the code also match the infinite multiplication factors k_{∞} and average scalar flux ratios for infinite multiplicative systems from the benchmark test sets.

The criticality results and the fixed source results from the Oklo code have been compared with criticality results and fixed source results from a discrete ordinates code and the results for both types of problems show a good agreement with the results from the discrete ordinates code as we increase the N for the discreet ordinates code.

Keywords: Integral Transport Equation, Collision Probability, Isotropic Scattering, Flat Flux Approximation, Discrete Ordinates

DECLARATION

I, the undersigned, hereby declare that the work contained in this project is my own original work.

_____

Sebenele Mugu Mtsetfwa

Date: 23rd May 2012

Johannesburg

ACKNOWLEDGEMENTS

I would like to extend my sincere gratitude to Dr. Oscar Zamonsky for guiding me through this journey. Thank you for putting in all the time, including your family time, and thank you for being a kind host during the countless visits to your home. Thank you for your patience with me when I couldn't follow some of the concepts. You have taught me to strive for quality; your refusal to be associated with mediocrity has produced this work and I am grateful for that attribute in you, I am proud to be associated with this type of work.

Contents

Figures

Tables

Abbreviations

This list contains the abbreviations used in this document.

Abbreviation or Acronym	Definition
CPU	Central Processing Unit
WNA	World Nuclear Association
Oklo	Name of the code developed in this project, named after the location in Gabon where fission was discovered to have occurred naturally.

1. PROJECT MOTIVATION

The quest for lower carbon emissions has resulted in a new nuclear renaissance, which has seen a large number for nuclear reactors being constructed worldwide. Nuclear energy accounts for about 15% of the total electricity generation in the world, and this share is expected to increase considerably with 65 reactors under construction [1]. The new growth comes with an increased need for reactor safety, to avoid incidents.

Nowadays, there is a great emphasis on safety of reactor designs from the experience gained on operating existing nuclear reactors. The new generation of reactors use evolutionary and passive safety measures. The evolutionary designs have improvements on safety systems from existing designs but still use active systems whilst the passive safety systems are designed to be reliant on natural phenomena to operate for example, gravity and heat transfer through natural means like convection, and conduction instead of relying on active systems like diesel engines which still have a chance to fail. There is an increased need for even more efficient reactor physics codes which are used for reactor design. Reactor physics codes are used to calculate amongst others the following variables:

- Neutron flux
- Neutron multiplication factor
- Power profile
- Reaction rates
- Control rod worth
- Critical dimensions

It is obviously practical to adjust these variables using a model until an optimum design is achieved than through experimental means, hence the importance of neutron transport codes in reactor design. The computational methods used for simulating and modelling neutron transport and interactions in the reactor core are either deterministic or stochastic in nature. The deterministic methods discretize the problem (space, angle, energy) resulting in a system of equations that are solved numerically [2]. Neutron transport codes are used to perform criticality safety calculations in support of design, development and licensing of nuclear installations.

Deterministic neutronics codes play a major role in reactor core modelling and simulation. The collision probabilities code in one dimension is useful in performing lattice calculations. This project will deliver a product that can be used to perform lattice calculations and it is also an academic exercise to help learn the design of a neutron transport code. The project develops a deterministic code to solve the integral transport equation using the method of Collision Probabilities.

1.1 Problem Statement

The project focuses on developing and implementing the method of collision probabilities to solve the integral Boltzmann transport equation. The integral transport equation is solved for criticality and fixed source problems.

1.2 Knowledge Gap to be filled

The project shall equip the student with the skills of developing deterministic neutron transport methods. This will enhance appreciation of the theory of reactor physics covered on the course work of the master's programme. The project shall also provide the university with a neutron transport method that can be used to train other students in designing deterministic neutron transport methods using different techniques.

1.3 Project Aims and Objectives

Develop a one-dimensional multigroup neutron transport code that solves the Boltzmann transport equation for multiplicative and non-multiplicative systems. The code will calculate the multiplication constants and flux distributions for fixed source problems. The code will be verified against some analytic benchmarks and some problems solved will be analyzed against a Discrete Ordinates code.

1.4 Structure of Report

The report is presented in 7 chapters with an addendum containing the structure of input files attached at the end of the report. Chapter 2 covers the literature review on the subject of this project. The literature review gives an over view of the transport equation. The review then covers different computational methods for solving the transport equation. The review culminates with the current state of methods with collision probabilities solvers. Chapter 3 discusses the theory on the subject matter that the project is based on. This section starts with a brief introduction of the integro-differential transport equation, followed by the integral transport equation. The method of the Collision Probabilities which is used to solve the integral transport equation is then discussed. Chapter 4 introduces the flow diagram on which the multigroup iteration scheme is based. In chapter 5 the results for different test cases executed in the project are presented, followed by a chapter that summarizes the results.

The conclusion of the work is presented in chapter 7 of the report.

2. LITERATURE REVIEW

This section introduces the literature that was consulted in this project on the neutron transport theory and methods development in this area. The section does not go into detail with the mathematical derivations of any of the concepts, with Section 3 of the report dealing extensively with the derivations.

2.1 Computational Methods for Neutron Transport

There are two classes of methods used to solve the neutron transport equation: deterministic and stochastic methods. This project develops a deterministic method that uses the method of Collision Probabilities to solve the integral form of the neutron transport equation. The different ways in which we treat the space and angular variables of the neutron transport equation yields different deterministic methods.

Lewis and Miller [2] gives a detailed discussion on the derivation of both the integro-differential transport equation and the integral transport equation. This book gives an introduction to the transport equation followed by the energy and time discretization of the transport equation. The energy discretization technique, which we will explain as applicable to the integral transport equation in the next Chapter of this report is used in several neutron transport methods. Amongst the neutron transport methods that are covered by Lewis and Miller [2] are the discrete ordinates, the collision probabilities and the Monte Carlo methods.

Stammler's book [3] offers a good insight into the numerical methods of steady state reactor physics. The book introduces the transport equation and then describes typical nuclear data libraries. The book then covers a number of numerical methods used to solve the transport equation, developing them with the underlying approximations in detail. Amongst these methods that are discussed is the Collision Probabilities method alongside the integral transport theory, the P_L approximations, the diffusion theory and the discrete ordinates method. The book also gives a good development of the multigroup iteration method and the flow diagram of this project's multigroup scheme was largely based on this work.

2.2 The Boltzmann Transport Equation

The roots of the transport theory dates back more than 100 years ago to the Boltzmann equation, first formulated for studying the kinetic theory of gases. With the advent of nuclear reactors in the 1940s, an interest in solving the neutron transport problem aroused [2]. Over the years, increasingly sophisticated numerical methods have been designed. The rapid decline in the cost of high computing power has also enhanced the design of powerful numerical methods without computing being the bottleneck. These methods are used to solve the neutron transport problem that is encountered in nuclear reactors and radiation shields in a multiregional and multidimensional form [2].

In deriving the neutron transport equation, only neutron interactions with the materials of the medium are considered. Neutron- neutron interaction can be safely neglected due to the low density of free neutrons compared with the atomic density of the media. The statistically large number of neutrons in a nuclear reactor allows averaging and the application of the linear Boltzmann equation [3].

The integral transport equation can be derived using first flight kernels to relate the angular flux in an element of phase space to the neutron emission rate due to fixed, scattering, and fission sources everywhere in the medium, and to sources on the boundary [4]. The result is not an explicit solution for the angular flux, but is an alternative form of the Boltzmann equation in an integral form. The integral transport equation leads to a treatment of the angular variable that is exact. Integral transport techniques were originally applied almost exclusively to the calculation of periodic flux distributions within the fuel-moderator-coolant cells of infinite reactor lattices because the highly absorbing regions and small spatial domains associated with such problem domains require a high-order angular approximation but relatively few spatial mesh points [2]. In this work, the integral transport equation for the angular flux is derived from physical definitions of neutronics quantities [5].The integral transport

equation will be obtained from the integro-differential equation in Section 3.2 of the report. From this equation we obtain the integral equation for scalar flux by simple integration on the angular variable.

The integral equation for scalar flux is then transformed to the multigroup integral transport equation which will be solved in the project for criticality eigenvalue and fixed source problems using the method of Collision Probabilities.

2.2.1 Monte Carlo Methods

In solving the neutron transport equation using deterministic computational methods, systematic computational errors are introduced by amongst other factors the discretization of the time, space, angle and energy and the representation of three-dimensional configurations [2]. In contrast, Monte Carlo methods are able to treat complex three-dimensional models. The continuous treatment of energy, space and angle removes the discretization errors that are associated with deterministic computational methods.

A Monte Carlo calculation on a high level consists of the following actions [2]:

- Simulating a finite number of particle histories through the use of pseudo random numbers. In each history the pseudo random numbers are used to track the length distances between collisions amongst other variables. Each history is begun by sampling the source distribution to determine the particle's initial positions energy and direction.
- The points of collision are determined using the mean free paths which are dependent on the material. By sampling the cross section data it can be determined with which nuclide the particle collided and what type collision occurred.
- These steps are repeated for a particle until it is absorbed or leaks from the system.

There are a number of production grade Monte Carlo codes in existence. Examples of such codes include MCNP, KENO Monte Carlo codes and TRIPOLLI4 Monte Carlo code to name but a few. Aragones J.M. et al [6] gives a good review of the existing core physics codes, both deterministic and Monte Carlo.

2.2.2 Deterministic Methods

The deterministic methods are based on discretizing the Boltzmann transport equation in each of the independent variables and solving the typically large system of algebraic equations that result. Lewis and Miller [2], Stammler [3] and The Nuclear Engineering Handbook [4] all give a detailed discussion of the different deterministic methods used to solve the transport equation.

The multigroup approximation method has been developed to discretize the energy variable in the neutron transport equation. The multigroup cross sections are determined so that significant reaction rates and/or leakages are preserved respect to the exact continuous energy problem. In this project we do not calculate cross sections but use them as a provided input from benchmark test sets. The application of the multigroup approximation to the integral transport equation is shown in Section 2 of this report.

The angular variable is generally discretized in discrete directions, e.g. the Discrete Ordinates (S_N) method, or in polynomial expansions as in the spherical harmonics (P_L) method. In this project the angular dependence is integrated out in the transport equation as will be shown in Section 2 of the report.

The spatial variable has probably been subjected to a greater variety of discretization methods than any other independent variable in the Boltzmann transport equation. Examples of common discretization techniques applied to the transport equation are the finite difference method (diamond difference and weighted diamond difference), finite element, nodal and characteristic methods. An important issue in the discretization of the space variable is the number of unknowns that must be calculated and stored per spatial cell [3]. Methods that require a minimum amount of storage are

generally less accurate on a specified grid, but the storage demand for neutron transport problems can be so high such that in many problems the simpler methods are preferred.

To improve the efficiency of both deterministic and probabilistic methods, some hybrid approximations of Monte Carlo and deterministic methods have been developed. For example the determination of some problem dependent biasing parameters which arises with variance reduction in Monte Carlo codes can be done efficiently with deterministic codes. Most available neutron transport codes are either deterministic or Monte Carlo, but there are a small number of hybrid codes which are now available [4]. For example the hybrid code MCBEND uses a multigroup diffusion solver to determine weight windows for Monte Carlo simulation [4]. The recent SCALE 6.0 package from Oak Ridge National Laboratory contains software which makes it possible to produce deterministically generated multigroup discrete ordinates solutions and turn them into weight windows for use in Monte Carlo simulations [4]. The Nuclear Handbook [4] explores the question of whether there is a class of hybrid methods for which Monte Carlo can be used to directly assist the accurate calculation of deterministic solutions. The main difficulty with deterministic solutions is the laborious calculation of multigroup cross sections. If a continuous energy Monte Carlo simulation could be used to calculate the problem dependent cross sections and supply this to the deterministic method, in this way the Monte Carlo method could significantly influence deterministic methods. The book cites some promising work that is being developed in this area. If such methods come to fruition then future transport methods will contain both Monte Carlo and deterministic modules where the Monte Carlo module supplies the multigroup cross section data to the deterministic module, and the deterministic module supplies biasing parameters to the Monte Carlo module. Such transport methods would be very close to black box systems, requiring minimal input from the user.

2.3 Solving the Integral Transport Equation using the Method of Collision Probabilities

The Collision Probability method is based on the integral form of the neutron transport equation. The main idea behind the integral transport method is to integrate out the angular dependence and to solve the neutron transport equation for the scalar flux directly [5]. The significant assumptions of the Collision Probability Method are the flat flux approximation and the isotropic scattering. The application of these assumptions in the development of the method will be shown in Section 3.2 of this report. The treatment of the spatial variables in the integral transport equation leads to dense matrices. This strong spatial coupling of large, dense matrices, which used to put a strong demand on computer memory and CPU time is no longer a serious problem with the advances in availability of computing power.

The Collision Probabilities method generally proceeds in three steps [4]:

1. A tracking process is applied over the lattice geometry to cover a sufficiently large number of neutron trajectories
2. The numerical calculation of the Collision Probabilities P_{ij} which involves integration over the angle as will be shown in Section 2 of the report. The Collision Probabilities have to be calculated for all energy groups.
3. Once the P_{ij} are known the flux can be computed.

For a problem containing N regions, the solution to the multigroup Collision Probabilities method produces a $N * N$ matrix in each energy group. Collision Probabilities can be defined over an infinite domain such as a lattice of identical cells. Reflective or periodic boundary conditions can be used to model the infinite system as will be shown in Section 3.4.5.2 of this work.

2.4 The Method of Characteristics

The method of characteristics (MOC) solves the integral transport equation for angular flux by following the straight neutron paths as the neutron moves across a domain [4]. This MOC is based on an iterative calculation of the particle flux by solving the transport equation of the neutron tracks

crossing the domain. The scalar flux per region and energy group is calculated by summing all mean angular fluxes from angular flux entering the domain and sources inside the domain.

It is interesting to note that the MOC has the same tracking information as the Collision Probability method. The MOC offers an alternative to the Collision Probability method and it overcomes the following limitations inherent in the Collision Probability method [4]:

- The Collision Probability method produces full square matrices of order equal to the number of regions in the domain whereas in the MOC the transport equation is solved for each of the neutron tracks through the domain which results in a fewer equations to solve. Memory and execution time requirements for the MOC increase linearly with the angular and spatial detail of the problem.
- The MOC can easily account for higher orders of anisotropy, whereas the Collision Probability method may have difficulties even with linear anisotropy.
- The MOC is preferred in cases where the number of regions exceeds a few hundred. The Collision Probabilities method would be preferred if one is interested in relatively strong local flux variations in a small system of the size of a pin cell.
- The MOC is usually more sensitive to the mesh size and may require a much finer discretization for the same accuracy as the Collision Probability method. Thin and small zones are undesirable because for certain azimuthal angles they are not covered by enough characteristics rays.
- The MOC is a good candidate for parallel computing to speed the execution time because the calculation for each characteristic track can be performed independently of other tracks.

It is beneficial to have both MOC and Collision Probability solvers in one code, in such cases the same geometrical description of the problem and track generating module can be used for both solvers.

2.5 The State of the Collision Probabilities Codes

There are a number of neutron transport codes which solve the neutron transport equation using the method of Collision Probabilities. What follows is a brief overview of a few codes with Collision Probability solvers [6]:

i. The DRAGON designed by the Ecole Polytechnique de Montreal Canada has an implementation of the Collision Probability method. The code has a 2 dimensional and 3 dimensional implementation of the Collision Probability method.
ii. APOLLO2 code has a one- dimensional and 2 dimensional implementation of the Collision Probability method.
iii. CASMO-5 code contains modules implementing the one-dimensional Collision Probability method and a two-dimensional method of characteristics implementation.
iv. The WIMS code, which consists of nuclear transport methods developed in the United Kingdom over a period of 30 years, has one and two-dimensional implementations of the Collision Probability method.
v. Another code which implements a Collision Probability solver is the HELIOS which has a 2 dimensional Collision Probability solver and a MOC solver in the latest version.

The list of Collision Probability solvers in existence discussed above is in no way exhaustive. This project aims to discuss the use of the Collision Probability methods to solve the integral transport equation in a clear and concise manner which is lacking in the available literature.

3. THEORY

This section covers the integro-differential transport equation, omitting the detailed derivation, the integral transport equation and it's the multigroup formalism.

3.1 The Integro-Differential Neutron Transport Equation

The solution of the neutron transport equation determines the distribution of neutrons in a system. In turn this distribution determines the rate at which various nuclear reactions occur within the system. The straight-line path of a neutron in a medium can be disturbed by the interaction of the neutron with the nuclei of the medium by different interactions: fission, capture and scattering to name a few.

The interaction of neutrons with nuclei of materials is governed by the concept of cross sections. The microscopic cross section $\sigma_x(E)$ is the effective cross- sectional area per nucleus seen by particles whilst the macroscopic cross section $\Sigma_x(E)$ is the probable number of reactions of type x per unit path length, and are both energy dependent.

The neutron transport equation is fundamental to reactor physics. The equation describes how the neutron distribution is established in a system with given cross section data. Only neutron interactions with the materials of the medium are considered. Neutron-neutron interaction can be safely neglected due to the low density of free neutrons compared with the atomic density of the media. This report is not giving the details of the derivation of the integro-differential form of the Boltzmann transport equation but only its description from the physics involved. The Boltzmann transport equation is given as

$$\left[\frac{1}{v}\frac{\partial}{\partial t}+\hat{\Omega}\bullet\nabla+\Sigma_t(\vec{r},E)\right]\psi(\vec{r},\Omega,E,t)=\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\Sigma_s(\vec{r},\hat{\Omega}'\bullet\hat{\Omega},E'\to E)\psi(\vec{r},\hat{\Omega}',E',t)$$
$$+\frac{\chi(E)}{4\pi}\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\nu(E')\Sigma_f(\vec{r},E')\psi(\vec{r},\hat{\Omega}',E',t) \quad , \qquad (3.0)$$

where:

$\psi(\vec{r},\hat{\Omega},E,t)$ = angular flux as function of space $\vec{r}$, energy E , angle $\hat{\Omega}$ and time t ,

$\Sigma_t(\vec{r},E)$ = total neutron macroscopic cross section,

$\Sigma_s(\vec{r},\hat{\Omega}'\bullet\hat{\Omega},E'\to E)$ = macroscopic scattering cross section form describing the transfer of particles with initial coordinates $E',\hat{\Omega}'$ before the interaction to $E,\hat{\Omega}$ after the interaction,

$\Sigma_f(\vec{r},E')$ = macroscopic fission cross section, and

$\chi(E)$ = fission neutron energy spectrum.

Nuclear data is treated as time independent because the time period over which any transient can occur is very much shorter than the time over which nuclear cross section data changes.

In this work we consider only the steady state transport equation. In this case, since the angular flux does not depend on the time variable, Eq. 3.0 is expressed as:

$$\left[\hat{\Omega}\bullet\nabla+\Sigma_t(\vec{r},E)\right]\psi(\vec{r},\hat{\Omega},E)=\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\Sigma_s(\vec{r},\hat{\Omega}'\bullet\hat{\Omega},E'\to E)\psi(r,\hat{\Omega}',E')$$
$$+\frac{\chi(E)}{4\pi k}\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\nu(E')\Sigma_f(\vec{r},E')\psi(\vec{r},\hat{\Omega}',E') \quad (3.1)$$

After eliminating the time dependence in the neutron transport equation as given by Eq. 3.1, the neutron transport equation is dependent on six variables namely: three position variables represented by $\vec{r}$, two direction variables represented by $\hat{\Omega}$ and the energy variable E .

It is not warranted that the system solved with Eq. 3.1 has a steady state solution. The criticality problem is formulated by assuming that ν , the average number of neutrons emitted per fission can be adjusted to obtain a time independent solution. To ensure that Eq. 3.1 has a steady state solution we modify the production term by dividing it by k . It can be shown that k is the measure of the number of neutrons in the current generation divided by the number of neutrons in the previous generation, which is defined as criticality. A reactor is critical if it has a self-sustaining neutron population without any external source, i.e., neutrons are only produced by neutrons in the system. In other words, physically a system containing fissionable material is said to be critical if there is a self-sustaining time independent chain reaction in the absence of external sources of neutrons [3].

A value of $k<1$ means that the hypothetical number of neutrons per fission $\frac{\nu}{k}$, required to obtain a stationary flux is larger than ν , the number of neutrons available per fission in reality. Therefore, such a system is sub critical and the modified fission rate has to be increased in order to obtain a non-trivial steady state solution for the neutron flux [2].

Conversely, if $k>1$, it means that fewer neutrons born in fissions are required to obtain a stationary flux than are currently produced in reality, a reduction in the fission rate is necessary to obtain a steady state solution. Therefore, such a system is supercritical. This can be summarized up as follows:

if $k>1$ the system is supercritical,

if $k=1$ the system is critical, and

if $k<1$ the system is sub – critical.

For a sub-critical system we can get a non-trivial solution to Eq. 3.1 by introducing an external neutron source. In this case, Eq. 3.1 can be written as:

$$\left[\hat{\Omega}\bullet\nabla+\Sigma_t(\vec{r},E)\right]\psi(\vec{r},\hat{\Omega},E)=\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\Sigma_s(\vec{r},\hat{\Omega}'\bullet\hat{\Omega},E'\to E)\psi(r,\hat{\Omega}',E')$$
$$+\frac{\chi(E)}{4\pi}\int_0^\infty dE'\int_{4\pi}d\hat{\Omega}'\nu(E')\Sigma_f(\vec{r},E')\psi(\vec{r},\hat{\Omega}',E')+Q(\vec{r},\hat{\Omega},E) \quad , \quad (3.2)$$

where $Q(\vec{r},\hat{\Omega},E)$ = is the external neutron source which emerge in the system from events other than fission.

If a neutron is inserted into a critical system, then after sufficient time has elapsed for the decay of transient effects, a time independent distribution of neutrons will exist in which the rate of neutron production is just equal to the rate of losses due to absorption and leakage from the system. If such an

equilibrium cannot be established, then the distribution of neutrons will either increase or decrease with time.

If neutrons from a time independent source are supplied to a sub critical system, the system will reach an equilibrium state characterized by stationary flux distribution in which the production rate from the external sources plus fission is in equilibrium with the absorption and leakage rates from the system.

But if the system is critical or supercritical no such equilibrium can exist in the presence of an external source and the neutron flux will be an increasing function of time.

When doing criticality calculations, i.e., calculating the value of the multiplicative constant k, we use Eq. 3.1 and modify the production term by dividing the number of neutrons born per fission by k, ν/k. In this project the integral form of the neutron transport equation is solved for both criticality and fixed source problems, with the detailed derivation shown in Section 3.4 of this report.

The effective multiplication factor k_{eff} gives the multiplication factor for a finite system whilst the infinite multiplication factor k_{∞} gives the multiplication factor for an infinite system.

3.2 The Integral Neutron Transport Equation

To obtain the integral form of the transport equation we will look first for one equation for the angular flux, $\psi(\vec{r},\hat{\Omega},E,t)$, i.e. neutrons with energy E that at the instant t are in $\vec{r}$ flying with direction $\hat{\Omega}$.

Let $q(\vec{r}',\hat{\Omega}',E,t')$ be the number of neutrons per $cm^3-\sec-eV-Steradian$ that are born at $\vec{r}'$ with energy E, direction $\hat{\Omega}'$ at time t', see Figure 1. Note that in order to arrive to the point $\vec{r}$ at time t, neutrons have to be born at time $t'=t-\frac{R}{v}$ at $\vec{r}'$, where v is the absolute value of their velocity, $\vec{v}=v\hat{\Omega}$. If all the neutrons are born in void where there are no collisions, all of them will contribute to $\psi(\vec{r},\hat{\Omega},E,t)$, therefore

$$d\psi(\vec{r},\hat{\Omega},E,t)=q(\vec{r}',\hat{\Omega},E,t-\frac{R}{v})dR. \tag{3.3}$$

If the medium is not void, the neutrons that arrive to $(\vec{r},\hat{\Omega},E,t)$ without colliding are

$$d\psi(\vec{r},\hat{\Omega},E,t)=q(\vec{r}',\hat{\Omega},E,t-\frac{R}{v})e^{-\Sigma R}dR, \tag{3.4}$$

where

$$\Sigma R\equiv\int_0^R dR'\Sigma_t(\vec{r}-\hat{\Omega}R',E)\equiv\tau(\vec{r},\vec{r}-R\hat{\Omega},E). \tag{3.5}$$

Taking into account the contribution of all neutrons along the trajectory from $\vec{r}'$ to $\vec{r}$,

$$\psi(\vec{r},\hat{\Omega},E,t)=\int_0^{\infty}dRq(\vec{r}',\hat{\Omega},E,t-\frac{R}{v})e^{-\tau(\vec{r},\vec{r}',E)}, \tag{3.6}$$

where the source is composed of an external source, scattering and fission contributions to the direction defined by $\hat{\Omega}$.

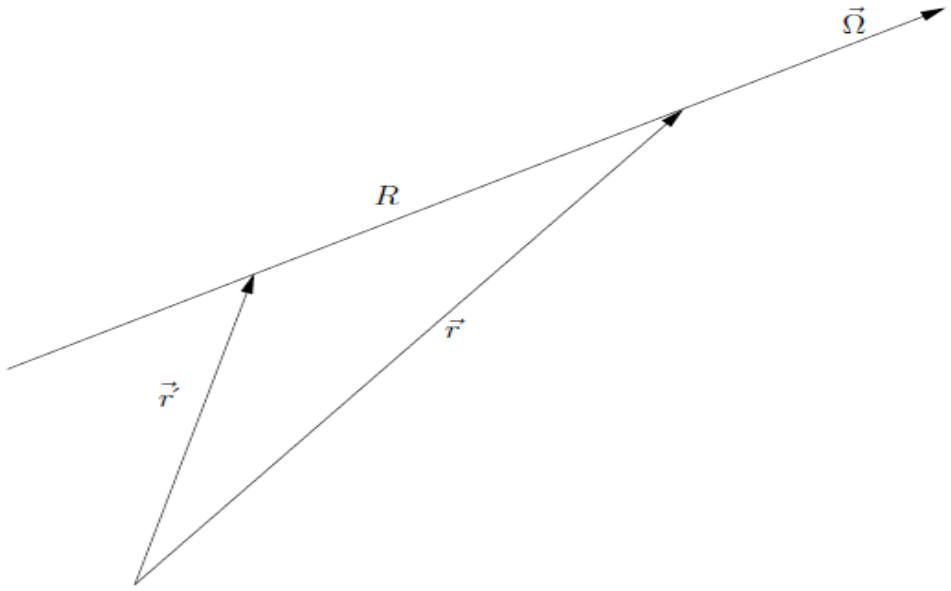


Figure 1: Path for Neutron Travel

In a finite system where R_s is the distance from $\vec{r}$ to the boundary along the direction $\hat{\Omega}$, the integral transport equation is expressed as

$$\psi(\vec{r},\hat{\Omega},E,t)=\int_0^{R_s} dRq(\vec{r}',\hat{\Omega},E,t-\frac{R}{v})e^{-\tau(\vec{r},\vec{r}',E)}+\psi(\vec{r}_s,\hat{\Omega},E,t-\frac{R_s}{v})e^{-\tau(\vec{r},\vec{r}',E)}, \quad (3.7)$$

Equation 3.7 is the integral form of the Boltzmann transport equation for the angular flux. The first term of the right hand side (RHS) of this equation can be interpreted as the non-collided contribution to the angular flux at $\vec{r}$ from all sources at $\vec{r}'$ contained in the straight line that goes from $\vec{r}$ to $\vec{r}-R_s\hat{\Omega}$ as shown in Figure 1. The uncollided neutrons that contribute to the angular flux at $\vec{r}$ from the incoming flux at the boundary of the system are given by the second term of the RHS of this equation. The angular flux is not any new "uncollided" type of flux but the usual one since the source term in Eq. 3.7 contains the scattering source which takes into account all collided neutrons that are redirected into $\hat{\Omega}$.

In this work, we solve Eq. 3.7 under the following assumptions:

1. Steady state, which means that $\psi(t)=\psi(t-\frac{R}{v})$,
2. Isotropic scattering, and
3. Isotropic external source

Under the last two assumptions, the source term which is composed of external, scattering and fission terms can be expressed as:

$$Q(\vec{r}',E)=q(\vec{r}',\hat{\Omega},E)=\frac{q^{ext}(\vec{r}',E)}{4\pi}+\int_0^{\infty} dE'\frac{\Sigma_s(\vec{r}',E'\to E)}{4\pi}\phi(\vec{r},E')+\frac{\chi(E)}{4\pi k}\int_0^{\infty} dE'\nu\Sigma_f(\vec{r},E')\phi(\vec{r},E'), \quad (3.8)$$

where $q^{ext}(\vec{r}',E)$ is the external source.

Then Eq. 3.7 can be written as

$$\psi(\vec{r},\hat{\Omega},E)=\int_0^{R_s} dRQ(\vec{r}-R\hat{\Omega},E)e^{-\tau(\vec{r},\vec{r}',E)}+\psi(\vec{r}_s,\hat{\Omega},E)e^{-\tau(\vec{r},\vec{r}',E)}. \quad (3.9)$$

In order to obtain the integral equation for the scalar flux, we multiply and divide the first term of the RHS of Eq. 3.9 by $R^2 = |\vec{r} - \vec{r}'|^2$ and integrate Eq. 3.9 in 4π,

$$\phi(\vec{r}, E) = \int_{4\pi} d\hat{\Omega} \int_0^{R_s} dR Q(\vec{r} - R\hat{\Omega}, E) e^{-\tau(\vec{r}, \vec{r}', E)} \frac{R^2}{|\vec{r} - \vec{r}'|^2} + \int_{4\pi} d\hat{\Omega} \psi(\vec{r}_s, \hat{\Omega}, E) e^{-\tau(\vec{r}, \vec{r}', E)} . \quad (3.10)$$

Noting that $d^3\vec{r}' = R^2 d\hat{\Omega} dR$, Eq. 3.10 can be written as

$$\phi(\vec{r}, E) = \int_V d^3\vec{r}' \frac{Q(\vec{r}', E)}{|\vec{r} - \vec{r}'|^2} e^{-\tau(\vec{r}, \vec{r}', E)} + \int_{4\pi} d\hat{\Omega} \psi(\vec{r}_s, \hat{\Omega}, E) e^{-\tau(\vec{r}, \vec{r}', E)} , \quad (3.11)$$

which is the integral equation for scalar flux. Note that $Q(\vec{r}', E)$ depends on the scalar flux, then, in the absence of incoming flux at the boundary at $\vec{r}_s$ or in an infinite system with no sources at infinite, Eq. 3.11 is an equation only in the scalar flux that is expressed as:

$$\phi(\vec{r}, E) = \int_V d^3\vec{r}' \frac{Q(\vec{r}', E)}{|\vec{r} - \vec{r}'|^2} e^{-\tau(\vec{r}, \vec{r}', E)} . \quad (3.12)$$

The next section of the report shows how to obtain the multigroup form of the previous equation. This form is the one solved in this work with the Collision Probability method.

3.3 Multigroup Formulation

In most numerical methods the neutron scalar flux is not treated as dependent on a continuous energy variable but the multi-energy group approximation. In our case, the scalar flux as a function of energy and space is the unknown to be obtained from Eq. 3.12. The methodology used in this work to solve the space dependence of the integral transport equation is the Collision Probabilities method which is explained in Section 3.4.1 of this report. The treatment of the energy variable with the multigroup approximation, which is common to all deterministic computational methods, is explained in this section.

Using the multigroup approximation the energy domain $(0, E_{\max})$ is divided into a number of disjoint intervals ΔE_g called energy groups, see Figure 2.

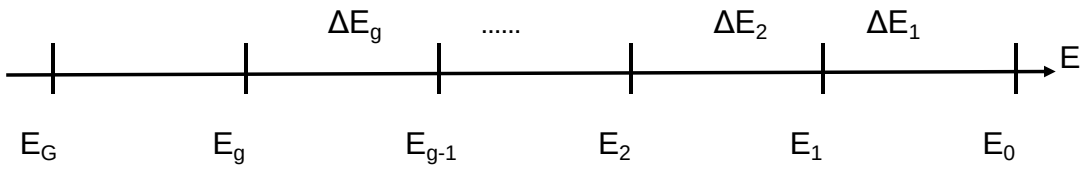


Figure 2: Division of the energy range into G energy groups

The subscripts on Figure 2 increase with decreasing energy which means that E_1 has a higher energy that E_2 for example.

The cross sections within each energy group are treated as constants i.e. equal to an average over the energy group. With the multigroup approximation applied the neutron transport equation becomes independent of energy within each group, then only dependent on space $\vec{r}$.

The equations that follow show the multigroup approximation applied to the integral transport equation, culminating with the appropriate multigroup formulation.

Integrating Eq. 3.12 in energy between the limits of group ΔE_g , i.e. E_g and E_{g-1} we get

$$\int_{E_g}^{E_{g-1}} \phi(\vec{r}, E)dE = \int_{E_g}^{E_{g-1}} dE \int_V d^3\vec{r}' \frac{Q(\vec{r}', E)e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi \mid \vec{r} - \vec{r}' \mid^2}. \qquad (3.13)$$

The integral of the scalar flux between E_{g-1} and E_g is defined as the total flux ϕ_g,

$$\phi_g(\vec{r}) \equiv \int_{E_g}^{E_{g-1}} \phi(\vec{r},E)dE\,, \tag{3.14}$$

being

$$\int_0^{\infty} \phi(\vec{r},E)dE = \sum_{g=1}^{G} \phi_g(\vec{r})\,. \tag{3.15}$$

Using Eq. 3.13, Eq. 3.14 can be written as

$$\phi_g(\vec{r}) = \frac{1}{4\pi}\int_V d^3\vec{r}\,' \frac{1}{|\vec{r}-\vec{r}\,'|^2} \int_{E_g}^{E_{g-1}} dE Q(\vec{r}\,',E) e^{-\tau(\vec{r},\vec{r}\,',E)}\,. \tag{3.16}$$

In the same way we can perform the integration involving the source term $Q(\vec{r}\,',E)$ with the source term given by Eq. 3.8. We will define for a group g the external source, fission and scattering terms respectively. In the same way, being the external source a density in the energy variable, the total external source in group g is defined as

$$\int_{E_g}^{E_{g-1}} q^{ext}(\vec{r},E)dE \equiv q_g{}^{ext}(\vec{r})\,. \tag{3.17}$$

The total average group cross section is defined such that the total reaction rate in the group is preserved, which gives

$$\Sigma_{t_g} \equiv \frac{1}{\phi_g}\int \Sigma_t(E)\phi(E)\,.$$

The exponential in Eq. 3.16 can be taken out of the integral by using the mean value theorem,

$$\tau_g(\vec{r},\vec{r}\,') \equiv \int_0^R dR'\Sigma_{tg}(\vec{r}-\hat{\Omega}R')\,, \tag{3.18}$$

then Eq. 3.16 becomes

$$\phi_g(\vec{r}) = \frac{1}{4\pi}\int_V d^3\vec{r}\,' \frac{e^{-\tau_g(\vec{r},\vec{r}\,')}}{|\vec{r}-\vec{r}\,'|^2} \int_{E_g}^{E_{g-1}} dE Q(\vec{r}\,',E)\,. \tag{3.19}$$

The integral over ΔE_g of the fission term is given by

$$\int_{E_g}^{E_{g-1}} dE\,\frac{\chi(E)}{4\pi k}\int_0^{\infty} dE'\nu\Sigma_f(\vec{r},E')\phi(\vec{r},E') = \frac{\chi_g}{4\pi k}\sum_{g'=1}^{G}\int_{E_{g'}}^{E_{g'-1}} dE'\nu\Sigma_f(\vec{r},E')\phi(\vec{r},E')\,, \tag{3.20}$$

where the fission spectrum over the group g is defined as

$$\chi_g \equiv \int_{E_g}^{E_{g-1}} dE\chi(E)\,. \tag{3.21}$$

Defining $\nu\Sigma_{fg}$ as the average value of $\nu\Sigma_f(E)$ in ΔE_g, using the same concept used for the total cross section, the fission source term can finally be expressed in the multigroup approximation as

$$\int_{E_g}^{E_{g-1}} dE \frac{\chi(E)}{4\pi k} \int_0^{\infty} dE' \nu\Sigma_f(\vec{r},E')\phi(\vec{r},E') \equiv \frac{\chi_g}{4\pi k} \sum_{g'=1}^{G} \nu\Sigma_{f_{g'}}(\vec{r})\phi_{g'}(\vec{r}) . \quad (3.22)$$

Integrating the scattering term from Eq. 3.8 within ΔE_g, it follows that

$$\frac{1}{4\pi} \int_{E_g}^{E_{g-1}} dE \int_0^{\infty} dE' \Sigma_s(\vec{r},E' \to E)\phi(\vec{r},E') = \frac{1}{4\pi} \sum_{g'=1}^{G} \int_{E_{g'}}^{E_{g-1}'} dE' \phi(\vec{r},E') \int_{E_g}^{E_{g-1}} dE \Sigma_s(\vec{r},E' \to E) , \quad (3.23)$$

which this gives

$$\frac{1}{4\pi} \sum_{g'=1}^{G} \int_{E_{g'}}^{E_{g-1}'} dE' \phi(\vec{r},E') \int_{E_g}^{E_{g-1}} dE \Sigma_s(\vec{r},E' \to E) \equiv \frac{1}{4\pi} \sum_{g'=1}^{G} \Sigma_{sg'g}\phi_{g'}(\vec{r}) . \quad (3.24)$$

The resulting source term as given by Eq. 3.8 under the multigroup approximation then becomes:

$$Q_g(\vec{r}') = \frac{1}{4\pi} \left(q_g^{\,ext}(\vec{r}) + \sum_{g'=1}^{G} \Sigma_{sg'g}\phi_{g'}(\vec{r}) + \frac{\chi_g}{k} \sum_{g'=1}^{G} \nu\Sigma_{fg'}(\vec{r})\phi_{g'}(\vec{r}) \right) . \quad (3.25)$$

Substituting for the multigroup approximation of the source term into Eq. 3.19 we arrive at the multigroup integral transport equation:

$$\phi_g(\vec{r}) = \int_V d^3\vec{r}' \frac{Q_g(\vec{r}')}{4\pi |\vec{r} - \vec{r}'|} e^{-\tau_g(\vec{r},\vec{r}')} . \quad (3.26)$$

The source term $Q_g(\vec{r}')$ in Eq. 3.26 which is given by Eq. 3.25 relates the flux in group g with fluxes from all other groups through the scattering and fission terms. With the use of the definitions, the transport equation which was dependent on both space and energy has now been written such that it is only dependent on space.

3.4 The Collision Probabilities Method

This section shows the derivation of the set of equations solved for average scalar flux with the scalar flux integral equation

$$\phi_g(\vec{r}) = \int_V d^3\vec{r}' \frac{Q_g(\vec{r}')}{4\pi\,|\,\vec{r}-\vec{r}'|} e^{-\tau_g(\vec{r},\vec{r}')}, \tag{3.27}$$

as the starting point, and the group source terms for criticality and fixed source calculations given by

$$Q_g(\vec{r}') = \frac{1}{4\pi}\left(\sum_{g'=1}^{G}\Sigma_{sg'g}\phi_{g'}(\vec{r}) + \frac{\chi_g}{k}\sum_{g'=1}^{G}\nu\Sigma_{fg'}(\vec{r}')\phi_{g'}(\vec{r})\right) \tag{3.28}$$

and

$$Q_g(\vec{r}') = \frac{1}{4\pi}\left(q_g^{\,ext}(\vec{r}) + \sum_{g'=1}^{G}\Sigma_{sg'g}\phi_{g'}(\vec{r}) + \chi_g\sum_{g'=1}^{G}\nu\Sigma_{fg'}(\vec{r}')\phi_{g'}(\vec{r})\right) \tag{3.29}$$

respectively.

3.4.1 Solution of the Integral Transport Equation Using the Collision Probabilities Method

The integral transport theory is used to perform calculations of reactor lattices, with the integral transport equation Eq. 3.12 as the starting point [7]. In this section we obtain the set of equations of the collision probabilities method for a system in void. The extension of collision probabilities to infinite systems is discussed in Sections 3.4.5.2.1 and 3.4.5.2.2 of this document.

The scalar flux integral equation, Eq. 3.26, is used as the starting point for the collision probabilities discussion and is repeated here:

$$\phi(\vec{r}) = \int_V d^3\vec{r}' \frac{Q(\vec{r}')}{4\pi\,|\,\vec{r}-\vec{r}'|^2} e^{-\tau(\vec{r},\vec{r}')}, \tag{3.30}$$

where the group subscript has been omitted for simplicity.

Defining

$$n(\vec{r}'\to\vec{r}) = \frac{e^{-\tau(\vec{r},\vec{r}')}}{4\pi|\vec{r}-\vec{r}'|^2}, \tag{3.31}$$

and stating the scattering term explicitly from the source term, Eq. 3.30 can be written as:

$$\phi(\vec{r}) = \int_V d^3\vec{r}'\, n(\vec{r}'\to\vec{r})[\Sigma_s(\vec{r}')\phi(\vec{r}') + Q(\vec{r}')]. \tag{3.32}$$

We note that $n(\vec{r}'\to\vec{r})$ is the non-collided flux at $\vec{r}$ due to an isotropic unit point source at $\vec{r}'$ and the source $Q(\vec{r}')$ term now contains fission and external contributions only.

In order to define the collision probabilities approximation to solve Eq. 3.26, the system is discretized into N disjoint volumes $V_i, i = 1...N$. Integrating Eq. 3.32 in the volume V_i and dividing by V_i, we get the average scalar flux in the region V_i given by

$$\phi_i \equiv \frac{1}{V_i}\int_{V_i} d^3\vec{r}\,\phi(\vec{r}) = \frac{1}{V_i}\int_{V_i} d^3\vec{r}' \int_{V} d^3\vec{r}'\, n(\vec{r}' \to \vec{r})[\Sigma_s(\vec{r}')\phi(\vec{r}') + Q(\vec{r}')]\,. \tag{3.33}$$

Since the set $\{V_i\}$ is disjoint, Eq. 3.33 can be expressed as

$$\phi_i = \frac{1}{V_i}\int_{V_i} d^3\vec{r} \sum_{j=1}^{N} \int_{V_j} d^3\vec{r}'\, n(\vec{r}' \to \vec{r})[\Sigma_s(\vec{r}')\phi(\vec{r}') + Q(\vec{r}')]\,. \tag{3.34}$$

We make the following assumptions:

1. The cross sections are constant within each region, i.e. $\Sigma_i = \Sigma(\vec{r})$ if $\vec{r} \in V_i$.
2. The flux is constant in each volume and is equal to its average value, $\phi_i = \phi(\vec{r})$ if $\vec{r} \in V_i$. This assumption is known as the flat flux approximation, FFA.
3. In each volume the source is constant and equal to its average value, $Q_i = Q(\vec{r})$ if $\vec{r} \in V_i$. This assumption is known as the flat source approximation, FSA.

Note that the FSA is only an approximation in the presence of external sources in the system, otherwise if there are no external sources the source term is only made up of reaction rates (fission, scattering), which are constant due to the FFA.

From these assumptions Eq. 3.34 can be written as

$$V_i\phi_i = \sum_{j=1}^{N}[\Sigma_{sj}\phi_j + Q_j]\int_{V_i} d^3\vec{r} \int_{V_j} d^3\vec{r}'\, n(\vec{r}' \to \vec{r})\,. \tag{3.35}$$

We define

$$P_{ij} = \frac{\Sigma_i}{V_j}\int_{V_i} d^3\vec{r} \int_{V_j} d^3\vec{r}'\, n(\vec{r}' \to \vec{r})\,, \tag{3.36}$$

then using this definition Eq. 3.35 can be written as

$$\Sigma_i V_i \phi_i = \sum_{j=1}^{N} V_j P_{ij}[\Sigma_{sj}\phi_j + Q_j]\,,\ i = 1,..., N\,. \tag{3.37}$$

Equation 3.37 gives a set of N equations, that are solved for N unknowns, $\phi_i, i = 1,..., N$, as functions of the cross sections, volumes and the coefficients P_{ij}. It is important to note that the integral transport equation has been transformed into a system of linear equations through the use of the three assumptions discussed previously.

Obtaining the solution for Eq. 3.37, and the different ways in which P_{ij} are calculated gives rise to the different collision probability methods.

The coefficients P_{ij} given by Eq. 3.36 are the first flight collision probabilities from region i to region j as they give the probability for a neutron born isotropically and uniformly distributed within region i

to have its first collision in region j. The first flight collision probability interpretation is clear if we note that Eq. 3.31 gives the uncollided flux at $\vec{r}$ due to an isotropic unit point source at $\vec{r}'$. Then, the integral over V_j of this kernel divided by the volume of region j gives the number of neutrons that arrive at $\vec{r}'$ without colliding from one neutron born isotropically and uniformly distributed in V_j. This number of neutrons multiplied by Σ_i and integrated over the volume V_i gives the total collision rate at V_i produced by one uncollided neutron born isotropically and uniformly distributed in V_j.

The solution to the integral transport equation using the method of collision probabilities improves in accuracy if the discretization is refined due to the flat flux approximation in the collision probabilities method.

From Eq. 3.37 it follows that for a domain with N meshes the resulting collision probabilities matrix is of size $N \times N$ per energy group. This matrix easily becomes very dense and expensive to solve and store. For example for a 6- energy group problem with 100 meshes in the domain a 100 by 100 matrix is solved at least 6 times without taking into account the number of outer iterations required before convergence of the multiplication factor for a criticality calculation.

The program developed in this project solves Eq. 3.37 for a one-dimensional slab geometry. It first calculates the first flight collision probabilities per energy group and stores them. The first flight collision probabilities are evaluated only once when a run is started because they are dependent on only material properties which do not change during multigroup iterations. Thereafter, the code calculates the sources per mesh and energy group due to the isotropic scattering and fission. Then the linear system of equations is solved for average scalar fluxes for the current energy group using Lower Upper (LU) matrix decomposition from the GNU Scientific Library [8]. The multiplication factor k_{eff} is then calculated for a criticality problem or the infinite multiplication factor k_{∞} calculated for an infinite system.

Note that P_{ij} are not independent. For example, in a system without leakages a neutron will suffer its first collision somewhere in the system, then

$$\sum_j P_{ij} = 1. \tag{3.38}$$

Neutron paths are reversible i.e. the path traversed by the neutron from i to j is the same path traversed by a neutron from j to i. Using this observation in Eq. 3.30 it follows that

$$n(\vec{r}_j \to \vec{r}_i) = n(\vec{r}_i \to \vec{r}_j). \tag{3.39}$$

If we exchange subscripts j for i we get the first flight collision probability P_{ji}. From Eq. 3.36 we can write expressions for $n(\vec{r}_j \to \vec{r}_i)$ and $n(\vec{r}_i \to \vec{r}_j)$, equating these expressions we get

$$\frac{V_j}{\Sigma_i} P_{ji} = \frac{V_i}{\Sigma_j} P_{ij} \Rightarrow \Sigma_i V_i P_{ij} = \Sigma_j V_j P_{ji}. \tag{3.40}$$

Equation 3.40 is known as the reciprocity relation. The reciprocity relation can be used to infer for example P_{ji} from some already calculated and stored P_{ij}, and this can result in a big memory saving for storage of the collision probability matrix if we apply the reciprocity relation to calculate P_{ji} for dense matrices as we need them instead of storing all of them, but in this project we are not yet using this property at this stage. This is one property of the collision probabilities that has been used to verify the collision probabilities results of the code developed for both void and reflective boundary conditions as shown in Section 5.1.

The next section shows the one-dimensional expressions corresponding to the generic equation for the first flight collision probabilities P_{ij} given by Eq. 3.36. These expressions are used in the code for evaluating the first flight collision probabilities for void reflective and periodic boundary conditions.

3.4.2 Expressions for First Flight Collision Probabilities for a Slab Geometry

The aim of this section is to derive an expression for the first flight collision probabilities in slab geometry expressed in terms of the exponential integral E_1 using Eq. 3.36.

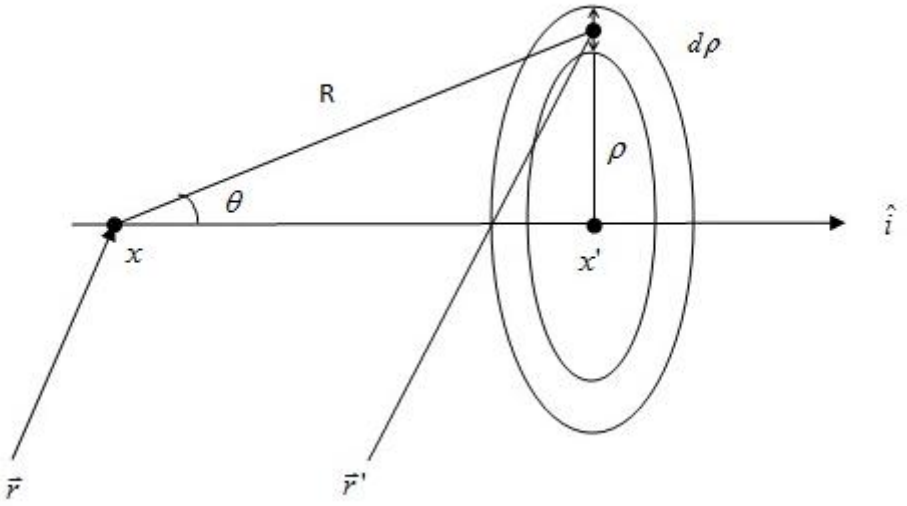


Figure 3: Spatial Discretization for Slab Geometry

Introducing Eq. 3.31 into Eq. 3.36 it follows that

$$P_{ij} = \frac{\Sigma_j}{4\pi V_i} \int_{V_i} d^3\vec{r} \int_{V_j} d^3\vec{r}' \frac{e^{-\tau(\vec{r},\vec{r}')}}{R^2}, \tag{3.41}$$

where the coordinates used can be seen in Figure 3.

Within the volume V_i, the differential volume can be expressed as $d^3\vec{r} = Adx$ where A is the transversal area to $\hat{i}$. Note that $V_i = A\Delta_i$ with Δ_i equal to the thickness of the slab i.

Based on Figure 3 we can write the differential volume where particles arrive as

$$d^3\vec{r} = 2\pi\rho d\rho dx \tag{3.42}$$

with R given by $R = |\vec{r} - \vec{r}'|$,and

$$\tau(\vec{r},\vec{r}') = \int_0^R \Sigma(s)ds\,. \tag{3.43}$$

Using the previous expressions in Eq. 3.41, it follows that

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_{\Delta i} dx \int_0^\infty \rho d\rho \int_{\Delta_j} dx' \frac{e^{-\tau(\vec{r},\vec{r}')}}{R^2} . \tag{3.44}$$

The following change of variables can be identified from Figure 3: $R^2 = |x - x'| + \rho^2$, then $RdR = \rho d\rho$. Using this change of variable Eq. 3.44 can be written as

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_{\Delta i} dx \int_{|x-x'|}^\infty \frac{dR}{R} \int_{\Delta_j} dx' e^{-\tau(\vec{r},\vec{r}')} . \tag{3.45}$$

To transform $\tau(\vec{r},\vec{r}')$ to the correct variables to perform the integration, we do a change of variables: $t = s\cos\theta \;\Rightarrow\; dt = \cos\theta ds$, where $\cos\theta = \frac{|x-x'|}{R}$ as can be seen from Figure 3. Then Eq. 3.43 can be written as

$$\tau(\vec{r},\vec{r}') = \int_0^R \Sigma(s)ds = \int_0^{R\cos\theta} \Sigma(t)\frac{dt}{\cos\theta} = \int_0^{|x-x'|} \Sigma(t)\frac{dt}{\cos\theta} = \frac{1}{\cos\theta}\tau(x,x') . \tag{3.46}$$

Substituting for $\tau(\vec{r},\vec{r}')$ as given by Eq. 3.46 in Eq. 3.45 we get

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_{\Delta_i} dx \int_{\Delta_j} dx' \int_{|x-x'|}^\infty \frac{dR}{R} e^{\tau(x,x')/\cos\theta} . \tag{3.47}$$

If we define a variable $\gamma = \frac{1}{\cos\theta} = \frac{R}{|x-x'|}$, it follows that $d\gamma = \frac{dR}{|x-x'|}$ and $\frac{dR}{R} = \frac{d\gamma}{\gamma}$.

Using the above mentioned expressions Eq. 3.47 can be expressed as

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_{\Delta_i} dx \int_{\Delta_j} dx' \int_{|x-x'|}^\infty \frac{d\gamma}{\gamma} e^{-\gamma\tau(x,x')} . \tag{3.48}$$

Using the definition of exponential integrals we can realize that the last integral in Eq. 3.48 corresponds to the exponential integral E_1 and the equation can be written as

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_{\Delta_i} dx \int_{\Delta_j} dx' E_1[\tau(x,x')], \tag{3.49}$$

which is the first flight collision probability expressed in terms of exponential integral E_1.

3.4.3 Derivation of Self Collision Probability Expression P_{ii} Slab Geometry

Equation 3.49 is used as the starting point for the derivation of the self-collision probability expression.

$$P_{ii} = \frac{\Sigma_i}{2\Delta_i} \int_{\Delta_i} dx \int_{\Delta_i} dx' \int_1^\infty \frac{d\gamma}{\gamma} e^{-\gamma\Sigma_i|x-x'|} \tag{3.50}$$

Due to the absolute value in the exponent in the last integral in Eq. 3.50, the integral is decomposed into 2 integrals depending on whether $x' > x$ or not giving:

$$P_{ii} = \frac{\Sigma_i}{2\Delta_i} \int_{\Delta_i} dx \int_1^\infty \frac{d\gamma}{\gamma} \left\{ \int_0^x dx' e^{-\gamma\Sigma_i(x-x')} + \int_x^{\Delta_i} e^{-\gamma\Sigma_i(x'-x)} \right\}. \tag{3.51}$$

Performing the integration in the curly brackets in Eq. 3.51 yields

$$P_{ii} = \frac{1}{2\Delta_i} \int_{\Delta_i} dx \int_1^\infty \frac{d\gamma}{\gamma^2} \left\{ e^{-\gamma\Sigma_i x}(e^{\gamma\Sigma_i x} - 1) + e^{\gamma\Sigma_i x}(e^{-\gamma\Sigma_i x} - e^{-\gamma\Sigma_i \Delta_i}) \right\}. \tag{3.52}$$

Multiplying out the exponents in Eq. 3.52 gives

$$P_{ii} = \frac{1}{2\Delta_i} \int_{\Delta_i} dx \int_1^\infty \frac{d\gamma}{\gamma^2} \left\{ 1 - e^{-\gamma\Sigma_i x} + 1 - e^{-\gamma\Sigma_i(\Delta_i - x)} \right\}. \tag{3.53}$$

Opening the curly brackets in Eq. 3.53 we get

$$P_{ii} = \frac{1}{2\Delta_i} \int_{\Delta_i} 2dx \int_1^\infty \frac{d\gamma}{\gamma^2} - \frac{1}{2\Delta_i} \int_{\Delta_i} dx \int_1^\infty \frac{d\gamma}{\gamma^2} \left[e^{-\gamma\Sigma_i x} + e^{-\gamma\Sigma_i(\Delta_i - x)} \right]. \tag{3.54}$$

The first part of Eq. 3.54 can be recognized as the exponential integral E_2, which can be evaluated to give: $E_2(0) = 1$. Integrating the exponents in x gives

$$P_{ii} = 1 - \frac{1}{2\Delta_i\Sigma_i} \int_1^\infty \frac{d\gamma}{\gamma^3} \left[1 - e^{-\gamma\Sigma_i\Delta_i} + e^{-\gamma\Sigma_i\Delta_i}(e^{\gamma\Sigma_i\Delta_i} - 1) \right], \tag{3.55}$$

which can be simplified further by using the properties for exponential integrals to get the final expression for evaluating self-collision probability for slab geometry

$$P_{ii} = 1 - \frac{1}{\Sigma_i\Delta_i} \left[\frac{1}{2} - E_3(\Sigma_i\Delta_i) \right]. \tag{3.56}$$

3.4.4 Derivation of First Flight Collision Probability Expression P_{ij} Slab Geometry

We consider a neutron born at point x' having its first collision at point x as shown in the Figure 4 below.

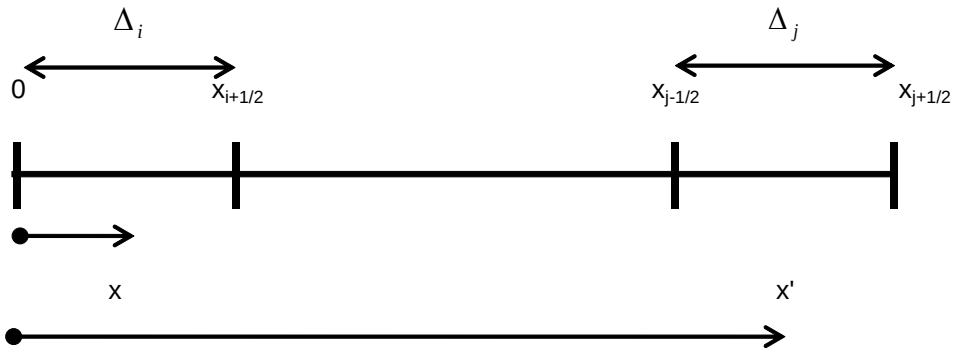


Figure 4: Discretization of slabs i and j

We will start the derivation with the same equation used for the derivation of the self-collision expression P_{ii} in the previous section:

$$P_{ij} = \frac{\Sigma_j}{\Delta_i} \int_{\Delta_i} dx \int_{\Delta_j} dx' \int_1^\infty \frac{d\gamma}{\gamma} e^{-\gamma\tau(x,x')} . \tag{3.57}$$

The optical distance between the two points x and x' is

$$\tau(x,x') = \Sigma_i (x_{i+1/2} - x) + \Sigma_j (x' - x_{j-1/2}) + \Sigma_{ij}\Delta_{ij} , \tag{3.58}$$

where $\tau_{ij} \equiv \Sigma_{ij}\Delta_{ij} \equiv \sum_{k-i+1}^{j-1} \Sigma_k \Delta_k$ and $\tau_k \equiv \Sigma_k \Delta_k$.

Substitute for $\tau(x,x')$ as given by Eq. 3.58 in Eq. 3.57, and use the limits for both Δ_i and Δ_j as shown in Figure 4 :

$$P_{ij} = \frac{\Sigma_j}{2\Delta_i} \int_0^{x_{i+1/2}} dx \int_{x_{j-1/2}}^{x_{j+1/2}} dx' \int_1^\infty \frac{d\gamma}{\gamma} e^{-[\gamma\Sigma_j (x' - x_{j-1/2}) + \gamma\Sigma_{ij}\Delta_{ij} + \gamma\Sigma_i (x_{i+1/2} - x)]} . \tag{3.59}$$

Integrating the first part and third part of the exponent of Eq. 3.59 in x' gives

$$\int_{x_{j-1/2}}^{x_{j+1/2}} dx' e^{-\gamma\Sigma_j (x' - x_{j-1/2})} = \frac{1}{\gamma\Sigma_j}\left[1 - e^{-\gamma\Sigma_j\Delta_j}\right], \tag{3.60}$$

and

$$\int_0^{x_{i+1/2}} dx e^{-\gamma\Sigma_i (x_{i+1/2} - x)} = \frac{1}{\gamma\Sigma_i}\left[1 - e^{-\gamma\Sigma_i\Delta_i}\right] \tag{3.61}$$

respectively.

Putting back the integration results into the Eq. 3.59 yields

$$P_{ij} = \frac{1}{2\Sigma_i \Delta_i} \int_1^\infty \frac{d\gamma}{\gamma^3} \left(1 - e^{-\gamma \Sigma_j \Delta_j}\right)\left(1 - e^{-\gamma \Sigma_i \Delta_i}\right) e^{-\gamma \Sigma_{ij} \Delta_{ij}} \,. \tag{3.62}$$

Then using the definition of exponential integrals on Eq. 3.62 the expression for evaluating the first flight collision probabilities for a slab geometry for $i \neq j$ is arrived at

$$P_{ij} = \frac{1}{2\Sigma_i \Delta_i} \left[E_3(\tau_{ij}) - E_3(\tau_{ij} + \tau_i) - E_3(\tau_{ij} + \tau_j) + E_3(\tau_{ij} + \tau_i + \tau_j)\right]. \tag{3.63}$$

3.4.5 Boundary Conditions on Evaluating Collision Probabilities

To do both criticality and fixed source calculations for finite and infinite systems we use the same method, but the difference is in the way we calculate the collision probabilities for the different boundary conditions. The reasoning for evaluating collision probabilities for void, reflective and periodic boundary conditions is shown in the sections that follow.

3.4.5.1 Void Boundary Condition

We consider a finite system with void boundary conditions on both sides as shown in Figure 5.

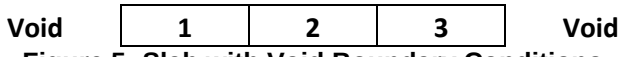


Figure 5: Slab with Void Boundary Conditions

To evaluate the first flight collision probabilities in this system we use either Eq. 3.56 or Eq. 3.63 depending on whether $j = i$ or $j \neq i$, because in deriving these equations we assumed that there are no contributions from outside.

3.4.5.2 Collision Probabilities for Infinite System

We need to evaluate the collision probabilities for an infinite system as shown in Figure 6.

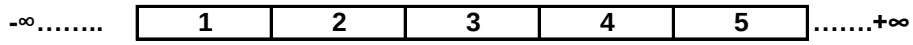


Figure 6: Infinite Slab System

For this system the first flight collision probabilities can be obtained as follows:

$$P_{ij} = \frac{\Sigma_{t_j}}{V_{i\infty}} \int_{V_{i\infty}} d^3\vec{r} \int_{V_{\infty j}} d^3\vec{r}' \frac{1}{4\pi |\vec{r} - \vec{r}'|^2} e^{-\tau(\vec{r},\vec{r}')} . \tag{3.64}$$

All regions i on all slabs are the same, i.e. we cannot distinguish one from the other so the total volume is the sum of all these volumes in all slabs making up the system,

$$V_{i\infty} = \sum_{N=1}^{\infty} V_i = \lim_{N\to\infty} NV_i . \tag{3.65}$$

On the other hand, the integral over $V_{i\infty}$ becomes

$$\int_{V_{i\infty}} d^3\vec{r}' = \lim_{N\to\infty} N \int_{V_i} d^3\vec{r}' . \tag{3.66}$$

The number of regions cancel and Eq. 3.64 can be written as

$$P_{ij} = \frac{\Sigma_{t_j}}{V_i} \int_{Vi} dv \int_{Vj\infty} dr' \frac{1}{4\pi |\vec{r} - \vec{r}'|^2} e^{-\tau(\vec{r},\vec{r}')} . \tag{3.67}$$

This equation is the similar to Eq. 3.36 except that the volume limit of region j extends from $-\infty$ to $+\infty$. So, in evaluating the first flight collision probabilities for an infinite slab lattice we need to cover

the domain over limits $-\infty$ and $+\infty$, the set of equations we use to evaluate the first flight collision probabilities are still Eq. 3.56 and Eq. 3.63. It is interesting to note that the collision probabilities expressed as in Eq. 3.67 correspond to the probability from *one* region *i* to *all* regions *j*.

To model an infinite system, we use reflective or periodic boundary conditions as will be discussed in the next two sections. The next section shows the generalization of how we locate the mesh index of interest in the extended slabs of the model in order to be able to calculate the collision probabilities for an infinite system. For the discussion on collision probabilities for infinite systems that follow we will refer to the first flight collision probability as P_{ij}^{*} to distinguish it from the collision probability for a finite system P_{ij}.

3.4.5.2.1 Evaluation of Collision Probabilities for Infinite Systems Using Reflective Boundary Condition

Figure 7 shows a lattice of slabs we have constructed using reflective boundary conditions to model an infinite system to calculate the collision probabilities. Once the collision probabilities have been evaluated the same code used to evaluate flux and the multiplication factor k_{eff} for criticality calculation is used to calculate group fluxes and the infinite multiplication factor k_{∞}.To model an infinite system as shown in Figure 6, we impose reflective boundary conditions on both ends of the system and get a model as shown in Figure 7.

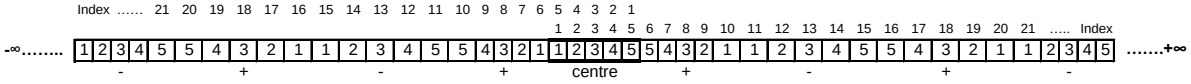


Figure 7: Example Infinite Slab Geometry Model Reflective BC.

To calculate the collision probabilities P_{ij}^{*} for an infinite system as modelled in Figure 7 we need to include contributions to P_{ij}^{*} due to collisions of a neutron born in mesh i and having its first collision in mesh j in the slab at the centre on which reflective boundary conditions have been imposed, plus all collision probabilities due to collisions by neutrons born in mesh i in the central slab and colliding for the first time in all meshes j in the extended slabs towards $-\infty$ and $+\infty$. We keep on adding the extensions of slabs towards a direction, $+\infty$ for example, until the contribution to the collision probability P_{ij}^{*} due to the collision in the mesh j in the extended slab is less than some required accuracy.

We need to distinguish whether a given extended slab is of even order or odd order to be able to decide where the mesh of interest j is located in the extended slab. We refer to a slab which is of odd count as odd parity and a slab of even count as even parity. For example, the first slab on the right of the central is slab number 1, which is odd, corresponding to +, and the second slab from the centre moving towards $+\infty$ is slab number 2 , which is even corresponding to -. The same concept is applied in identifying the order of slabs from the centre to $-\infty$.

To calculate P_{11}^{*} for example (based on Figure 7), the collision probabilities due to collisions from $i=1$ to meshes j towards $+\infty$ we have:

$$P_{11}^{*} = P_{11} + P_{110} + P_{111} + P_{120} + P_{121}.$$

The contributions to P_{11}^{*} due to collisions from a neutron born in mesh $i=1$ and having its first collision in any of the meshes j in the extended slabs towards $-\infty$ is

$$P_{11}^{*} = P_{56} + P_{515} + P_{516} + P_{524},$$

where we have omitted the collision probability P_{11} due to self-collision in the slab at the centre which has already been included in the $+\infty$ sweep. We will use the equation for P_{ii}, Eq. 3.56 for the self-collision probability in the slab at the centre but for all other collisions from 1 to all meshes j in the extended slabs we use the equation for P_{ij}, Eq. 3.63 to calculate the first flight collision probability.

We can use the foregoing discussion to come up with a general method on how to find the location of the mesh j in the extended slab in order to calculate P_{ij} to add to P_{ij}^{*} by using the slab parity concept.

Centre to $+\infty$ Case

For the iteration of the contribution from j in the central slab, during the first run j is in the central slab so we just do the equivalence test ($j=i$ or $j\neq i$) and use the appropriate equation to compute the collision probabilities.

The problem is in finding the location of j in the extended slabs. We keep track of what j is in the slab we start with at the centre of Figure 7, for example when computing the collision probability P_{12}, $j=2$, and we need to find its index on the extended slabs.

- If slab extension is of odd parity, j is at new lattice length minus index of j location in slab at centre, which gives index number 9 for the first extended slab based on the Figure 7.
- If slab extension is of even parity, j is at the same mesh index in the extended slab under consideration as it is in the central slab, which gives index number 12 on the second extended slab on Figure 7.

Centre to $-\infty$ Case

The location of j on the slabs on the left hand side (LHS) of the centre is the opposite of the generalization for the RHS case, i.e.:

- If slab extension is of odd parity, j is at the same mesh index in the extended slab under consideration as it is in the central slab. For example, for P_{12} towards $-\infty$, the index of j in the first extended slab is index number 7.
- If slab extension is of even parity, j is at new lattice length minus index of j location in central slab. The location of $j=2$ on the second extended slab is index number 14.

When all the contributions have been added to get P_{ij}^{*} the condition $\sum_{j} P_{ij} = 1$ should hold. With the collision probabilities calculated the same code used for criticality calculations is used to calculate k_{∞} and group scalar fluxes with such results shown in Section 5.5 of this report.

3.4.5.2.2 Evaluation of Collision Probabilities for Infinite System Using Periodic Boundary Conditions

We can use periodic boundary conditions to model an infinite system. We consider an infinite system discretized into 3 meshes for which we need to calculate the Collision Probabilities. Such a system can be modelled with periodic boundary conditions as shown in Figure 8.

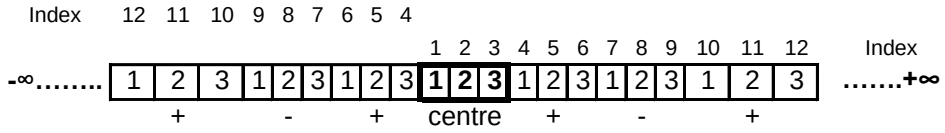


Figure 8: Infinite Periodic System Model

The slab at the centre represents the system on which we impose reflective boundary conditions on both ends. The location of a mesh j is always at the same location it is in the central slab for the periodic slab lattice.

For example to calculate P_{11}^{*}, we will calculate the first contribution due to self-collision in the central slab using the self-collision probability equation Eq. 3.56. Then we calculate the collision probability due to collisions in all meshes j in the extended slab towards $+\infty$. The index of j in the extended slab is in the same position as j in the central slab. For the example under consideration it is at mesh number 4, which is the first mesh of the extended slab. The same reasoning is applied for all slabs extended towards $+\infty$ until the convergence criteria is reached.

To calculate the contributions to P_{11}^{*} due to collisions occurring in meshes j located in slabs extended on the left of the central slab towards $-\infty$, we also need to figure out the location of the index of the mesh of interest j in the extended slab. In the extended slab towards the left j will always be located at the length of the central slab plus the length of all extended slabs minus the index of the location of j in the central slab. In our example for calculating P_{11}^{*} j is in index number 6 on the first extended slab, which is the length of the slab at the centre and the first extension minus zero, which is the index where 1 is located in the slab at the centre (see Figure 8).

Like we did for infinite systems with reflective boundary conditions on the previous section, we need to ensure that we include the contribution to the collision probability P_{ij}^{*} due to the collisions in meshes in the central slab only once as we calculate the contributions to P_{ij}^{*} due to collisions taking place in the extended slabs.

4. SOFTWARE FLOW DIAGRAMS AND EQUATIONS

This section presents the set of equations programmed, the main flow diagram used to design the software and the expressions evaluated at each process block of the flow diagram.

4.1 Set of Equations

The following sets of equations are programmed in the code to evaluate the first flight collision probabilities:

$$P_{ii} = 1 - \frac{1}{\Sigma_i \Delta_i}\left[\frac{1}{2} - E_3(\Sigma_i \Delta_i)\right] \tag{4.0}$$

$$P_{ij} = \frac{1}{2\Sigma_i \Delta_i}\left[E_3(\tau_{ij}) - E_3(\tau_{ij} + \tau_i) - E_3(\tau_{ij} + \tau_j) + E_3(\tau_{ij} + \tau_i + \tau_j)\right] \tag{4.1}$$

With the first flight collision probabilities evaluated the linear system of equations is programmed from Eq. 3.37, repeated here:

$$V_j \Sigma_{t_j} \phi_j = \sum_{i=1}^{N} V_i P_{ij} (Q_i + \Sigma_{sgg_i} \phi_i)\,. \tag{4.2}$$

The source density term for the external source problems is given by

$$Q_g(\vec{r}) = \frac{1}{4\pi}\left(q_g{}^{ext}(\vec{r}) + \sum_{g'=1}^{G} \Sigma_{sg'g} \phi_{g'} + \chi_g \sum_{g'=1}^{G} \nu\Sigma_{fg'}(\vec{r}) \phi_{g'}(\vec{r}) \right), \tag{4.3}$$

whilst source density term for the criticality problems is given by

$$Q_g(\vec{r}) = \frac{1}{4\pi}\left(\sum_{g'=1}^{G} \Sigma_{sg'g} \phi_{g'} + \frac{\chi_g}{k} \sum_{g'=1}^{G} \nu\Sigma_{fg'}(\vec{r}) \phi_{g'}(\vec{r}) \right). \tag{4.4}$$

Equations 4.0 and 4.1 give the first flight collision probabilities. Equation 4.3 and Eq. 4.4 are used to formulate the source per mesh for the energy group being evaluated. The matrices resulting from the set of linear equations formulated from Eq. 4.2 are solved using LU decomposition from the open-source GNU Scientific Library [5].

4.2 Flow Diagram Multigroup Iteration

Figure 9 shows the multigroup iteration scheme for criticality calculations on which the code design has been based; the equations solved follow the diagram.

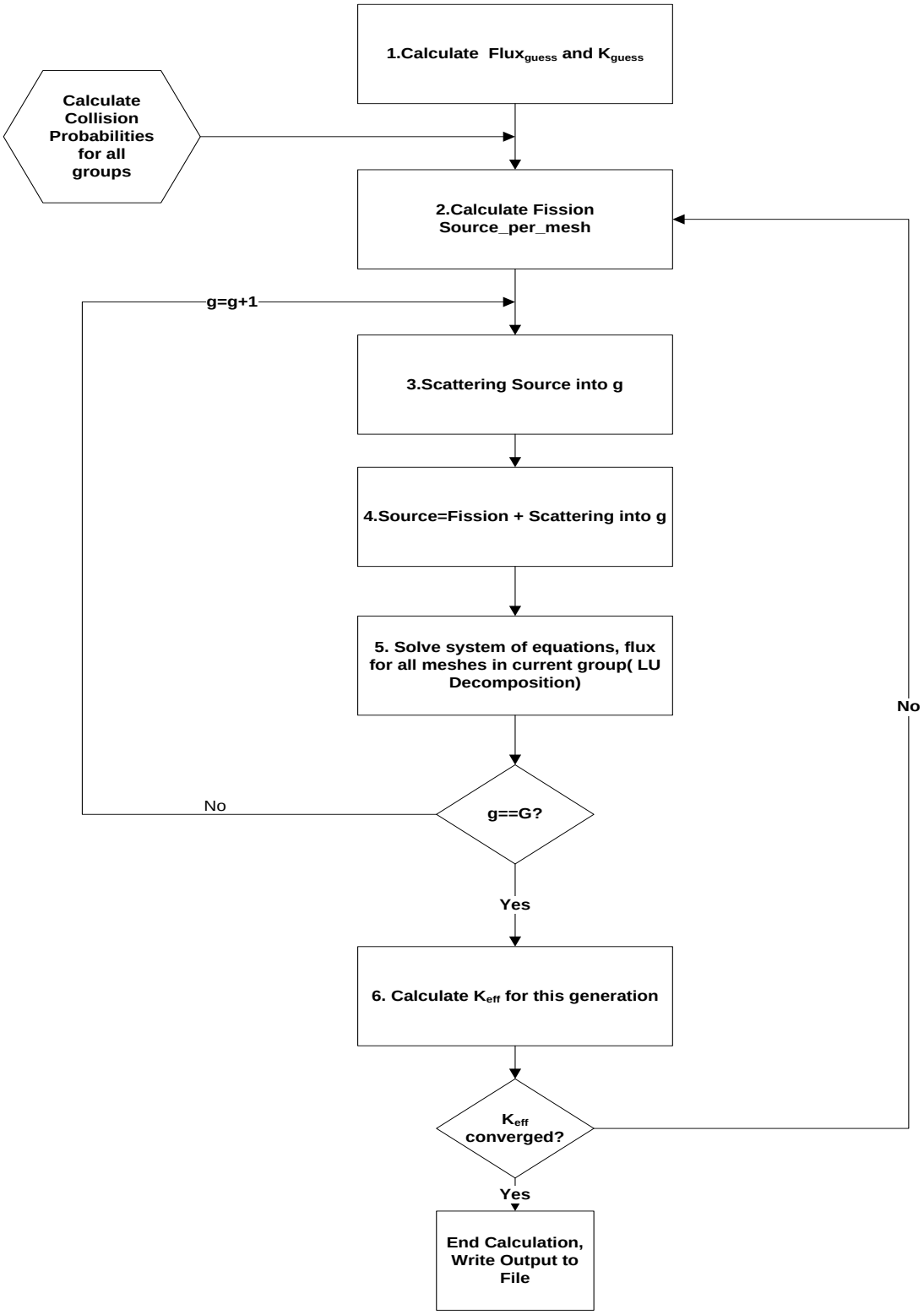


Figure 9: Flow Diagram Criticality Calculation

Each of the iterations starts with the calculation of the guess flux and the fission source from the guessed value of the multiplication factor k. Numerically, the division of the fission source in the current generation t by the fission source of the previous generation is just a normalization which ensures that the amplitude remains at a fixed value [3]. Therefore, the group fluxes of the subsequent generations are normalized to the same fission source level. The amplitude of the fission source can be normalized arbitrarily, for example if we normalize it to 1 fission neutron born in the system at time $t-1$, the multiplication factor for generation t is found to be

$$k = \sum_i V_i \sum_g \nu\Sigma_{fig}\phi_{ig} \ . \tag{4.5}$$

The guess flux for the previous generation where the multiplication factor was normalized to 1, is given by

$$\phi_{guess} = \frac{1}{\sum_i V_i \sum_g \nu\Sigma_{fig}} , \tag{4.6}$$

and from this it follows that the fission source per mesh for the first generation is given by

$$FS_i = \frac{1}{k}\sum_g \nu\Sigma_{fig}\phi_{guess} \ . \tag{4.7}$$

The guess flux is constant for all meshes in all energy groups. At the start of the program we have all the variables we need to calculate the guess flux, which are all material dependent.

Using the source term density Eq. 4.4, we get the expression for fission source per mesh (FS_i) for subsequent generations.

$$FS_i = \frac{1}{k}\sum_g \nu\Sigma_{fig}\phi_{ig} \ . \tag{4.8}$$

The fission source per mesh for subsequent generations is calculated using the multiplication factor for the previous generation. The scattering source into the current group excluding self-scattering is given by:

$$Source\_Scattering = \sum_{g'\neq g}\Sigma_{sg'g}\phi_{g'i} \ . \tag{4.9}$$

The collision probabilities are calculated using the set of equations Eq. 4.0 and Eq. 4.1 for void, periodic and reflective boundary conditions. The collision probabilities are evaluated once and stored for all energy groups at the start of the multigroup iteration.

Equation 4.2 is solved by LU matrix decomposition in step 5 of the flow diagram

The source Q_i into the current group g is obtained by combining the fission source from step 3 and the scattering source from step 4 of the flow diagram.

The calculation of flux and multiplication factor for the fixed source problem follows the same flow diagram shown in Figure 9, but the systems under consideration must be sub-critical with an external source introduced. The calculation converges if the differences in the scalar flux between generations are smaller than the specified epsilon.

The expression used to evaluate the convergence of the outer iteration is $\left|k^t - k^{t-1}\right| \leq \varepsilon$.

where the value of the convergence has been set to 1E-06 for all test cases solved in this work. The convergence criteria for the outer iteration for fixed source problems is similar to the one for criticality

calculation problems but we evaluate the convergence in flux for the current iteration compared with the previous one instead of the multiplication factors.

5. CODE VERIFICATION RESULTS AND DISCUSSION

The flow diagram on Figure 10 shows the steps involved in generic computer simulation of a physical phenomenon that are applied in developing radiation simulation [9]. The diagram shows the relationship between physical phenomena, computational model, verification and validation.

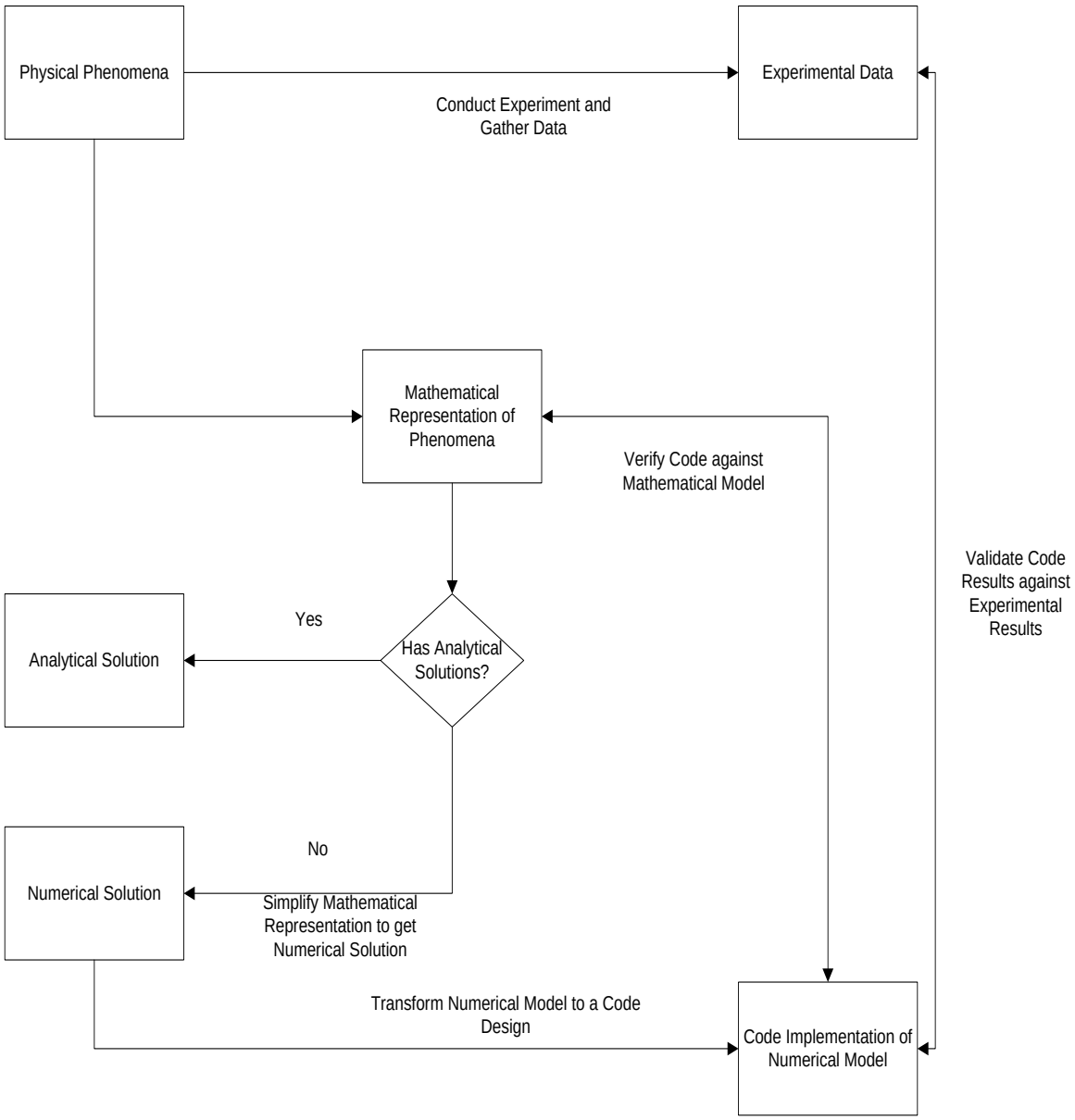


Figure 10: Computer Simulation of Physical Phenomena

The diagram shows a summary of stages involved in developing a computer simulation of radiation transport. The first exercise in developing the simulation is to develop a theoretical physics model and describe it with mathematical expressions, which corresponds to the derivation of the integral transport equation, the multigroup formalism that follows up to the linear systems of equations and the collision probabilities approach in this project. The solution to the resulting expressions is computed using the developed code Oklo in this project.

Verification is the connection between the resulting mathematical equations and the code that seeks to solve the equation. Through verification we compare the results calculated by the computer code with the analytical solutions of the mathematical equations describing the neutron transport phenomena. Verification may also be performed against results from other computer codes.

Verification is defined as the process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of the phase [9] in other words it is a proof of correctness.

A closely related concept is validation, which is the process of evaluation of a system at the end of the development phase to ascertain whether it satisfies the specific requirements. Validation compares the output calculated by the software with experimental results. The diagram shows that there is no direct link between a physical phenomenon and a computer code, these two are linked by a mathematical model of the physical phenomena under consideration.

In this work, the collision probabilities have been checked against the reciprocity relation, the condition that $\sum_j P_{ij} < 1$ for void boundary conditions and $\sum_j P_{ij} = 1$ for reflective boundary conditions, where $j = 1, \ldots, N$ and N is the total number of meshes over the domain. The criticality calculations and infinite lattice calculations are verified against analytical benchmarks for criticality codes from Los Alamos National Laboratory [9] that have been used to verify MCNP amongst other codes. These benchmarks have been gathered from peer reviewed journals.

The flux for all results presented in this work is of arbitrary units because the power has not been fixed for all the problems.

Notation for Test Cases

The notation for test cases is the same as in [9] but it is reproduced here for clarity reasons. The naming convention used for all bare geometry test cases is: Fissile Material- Energy Groups- Scattering- Geometry according to the nomenclature described in Table 1. The fissile material identifiers used are:

1. PU for Pu-239,
2. U for U-235,
3. UAL for a homogeneous mixture of highly enriched uranium and aluminium,
4. UD2O for low enriched uranium homogenously mixed with D_2O, and
5. URR for 93% enriched uranium used in research reactors.

The naming convention for multi-media cases is Fissile Material-Reflecting Material (thickness)-Energy Groups-Scattering-Geometry.

The possible entries are given in the nomenclature table:

Table 1: Nomenclature for Test Cases

Fissile Material	Energy Groups	Geometry	Reflector Material
PU	1 group	**IN**finite	Bare
U	2 groups	**SL**ab	H_2O
UD2O	3 groups	**I**nfinite **S**lab **L**attice **C**ell	
UAL	6 groups		
URR			

The following examples show the use of the notation explained before:

U-2-0-SL is a test case with fissile material U-235, 2 energy groups, isotropic scattering and slab geometry.

UAL-2-0-IN corresponds to an infinite (no reflector) homogeneous system composed of enriched uranium and aluminium, 2 energy groups and isotropic scattering.

PUa-H2O (1)-1-0-SL is a slab reactor with Pu-239, reflected with water of 1 mean free path thickness, one energy group and isotropic scattering.

URRd-H2O(10-2-0-ISLC is two group infinite slab lattice cell system with a water reflector of 1 mean free path.

5.1 Verification of Collision Probabilities for systems with void boundary conditions

At the current stage of the development of the code the collision probabilities are calculated using the expressions for self-collision probabilities given by Eq. 4.0 and Eq. 4.1 for all meshes and energy groups. The reciprocity relation $V_i\Sigma_{ti}P_{ij} = V_j\Sigma_{tj}P_{ij}$ has been verified for each mesh and energy group.

For a finite system composed of N meshes with void boundary conditions, the sum $\sum_j^N P_{ij}$ must be less than one for all mesh i due to leakages from the system. This condition has been verified in the cases where it applies.

In an infinite system modelled as having N meshes surrounded by reflective boundary conditions there are no leakages, then $\sum_j^N P_{ij}$ must be equal to one for all mesh i. This relation has been also verified in the infinite lattice cases solved.

As an example, Table 2 and Table 3 show the collision probability matrices calculated for a homogeneous slab immersed in void and under reflective boundary conditions respectively with a 4 mesh uniform discretization. It can be noted that all previously discussed conditions are met.

Table 2: Collision Probabilities for a finite system composed of Pu-239(a)

P_{ij}	1	2	3	4
1	0.33513	0.15278	0.07097	0.03995
2	0.15278	0.33513	0.15278	0.07097
3	0.07097	0.15278	0.33513	0.15278
4	0.03995	0.07097	0.15278	0.33513
Total	0.59883	0.71166	0.71166	0.59883

Table 3: Collision Probabilities for an infinite system composed of U-235(b)

P_{ij}	1	2	3	4
1	0.59954	0.23412	0.10163	0.06471
2	0.23412	0.46707	0.19718	0.10163
3	0.10163	0.19718	0.46707	0.23412
4	0.06471	0.10163	0.23412	0.59954
Total	1.00000	1.00000	1.00000	1.00000

5.2 Criticality Calculation Results

This section shows some criticality calculations performed with the code compared with the benchmark test cases.

Table 4 shows the list of the criticality problems that are solved for finite systems while Table 5 presents the cases corresponding to infinite systems. Both Table 4 and Table 5 must be interpreted using the key in the nomenclature on Table 1.

Table 4: List of Test Cases for Criticality Calculations in Finite Systems

Test Case Type	Test Case Name	Section in Report
One medium one energy group	Pub-1-0-SL	5.2.2
	Ua-1-0-SL	5.2.2
	UD2O-1-0-SL	5.2.2
Two media one energy group	PUa-H2O(1)-1-0-SL	5.2.2
	PUa-H2O(0.5)-1-0-SL	5.2.2
	UD2O-H2O(1)-1-0-SL	5.2.2
	UD2O-H2O(10)-1-0-SL	5.2.2
One medium two energy groups	PU-2-0-SL	5.2.2
	U-2-0-SL	5.2.2
	UAL-2-0-SL	5.2.2
	URRa-2-0-SL	5.2.2
	UD2O-2-0-SL	5.2.2
Two media two energy groups	URRb-H2Oa(1)-2-0-SL	5.2.2
	URRb-H2Oa(5)-2-0-SL	5.2.2
One medium six energy groups	U-6-0-SL	5.2.3
Two media six energy groups	U-H2O-6-0-SL	5.2.4

Table 5: List of Test Cases for Criticality Calculations in Infinite Slab Lattices

Test Case Type	Test Case Name	Section in Report
One medium one energy group	PUa-1-0-IN	5.5.1
	Pub-1-0-IN	5.5.1
	Ua-1-0-IN	5.5.1
	Ub-1-0-IN	5.5.1
	Uc-1-0-IN	5.5.1
	Ud-1-0-IN	5.5.1
	UD2O-1-0-IN	5.5.1
	Ue-1-0-IN	5.5.1
One medium two energy groups	PU-2-0-IN	5.5.2
	U-2-0-IN	5.5.2
	UAL-2-0-IN	5.5.2
	URRa-2-0-IN	5.5.2
	URRb-2-0-IN	5.5.2
	URRc-2-0-IN	5.5.2
	URRd-2-0-IN	5.5.2
	UD2O-2-0-IN	5.5.2

5.2.1 Cross section Data for Benchmark Test Cases

This section presents the cross section data for benchmark test cases which were used in this project for verification of the criticality calculations of the Oklo code for both finite and infinite systems. The test cases considered here can be categorized into: one medium-one energy group, two media-one energy group, one medium-two energy group and two media-two energy group test cases.

Table 6: Cross Section Data for One Medium One Energy Group Cases

Material	ν	$\Sigma_f\ (cm^{-1})$	$\Sigma_s\ (cm^{-1})$	$\Sigma_t\ (cm^{-1})$
Pu(b)-239	2.84	0.081600	0.225216	0.32640
U(a)-235	2.70	0.065280	0.248064	0.32640
UD2O	1.70	0.054628	0.464338	0.54628

Table 7: Cross Section Data for Two Media One Energy Group Cases

Test Case Name	Material Name	ν	$\Sigma_f\ (cm^{-1})$	$\Sigma_s\ (cm^{-1})$	$\Sigma_t\ (cm^{-1})$
PU-H20(0.5)-1-0-SL, PU-H20(1)-1-0-SL	Pu-239	3.24	0.081600	0.225216	0.32640
	H_2O	0.0	0.0	0.032640	0.293760
UD2O-H2O(1)-1-0-SL, UD2O-H2O(10)-1-0-SL	UD_2O	1.70	0.054628	0.464338	0.54628
	H_2O	0.0	0.0	0.491652	0.54628

Table 8: Cross Section Data for One Medium Two Energy Group Test Cases

Material Name	Energy Group	ν_1	$\Sigma_{f1}\ (cm^{-1})$	$\Sigma_{11}\ (cm^{-1})$	$\Sigma_{12}\ (cm^{-1})$	$\Sigma_{t1}\ (cm^{-1})$	χ_1
Pu-239	Fast	3.10	0.0936	0.0792	0.0432	0.2208	0.575
	Thermal	2.93	0.08544	0.0	0.23616	0.2208	0.425
U-235	Fast	2.70	0.06192	0.078240	0.078240	0.2160	0.575
	Thermal	2.50	0.06912	0.0	0.26304	0.3456	0.425
U-Al	Fast	0.0	0.0	0.247516	0.020432	0.268165	1.0
	Thermal	2.830023	0.060706	0.0	1.213127	1.276976	0.0
URR	Fast	2.50	0.0010484	0.62568	0.029227	0.65696	1.0
	Thermal	2.50	0.050632	0.0	2.44383	2.52025	0.0
UD2O	Fast	2.50	0.002817	0.31980	0.0045552	0.33588	1.0
	Thermal	2.50	0.097	0.0	0.42410	0.54628	0.0

Table 9: Cross section Data for URRb and H2O Reflector Fast Energy Group

Material Name	ν_1	Σ_{f1} (cm^{-1})	Σ_{11} (cm^{-1})	Σ_{12} (cm^{-1})	Σ_{t1} (cm^{-1})	χ_1
URRb	2.50	0.000836	0.83892	0.4635	0.88721	1.0
H_2O(a)(Reflector)	0.0	0.0	0.83975	0.04749	0.88798	0.0

Table 10: Cross section Data for URRb and H2O Reflector Thermal Energy Group

Material Name	ν_2	Σ_{f2} (cm^{-1})	Σ_{21} (cm^{-1})	Σ_{22} (cm^{-1})	Σ_{t2} (cm^{-1})	χ_2
URRb	2.50	0.029564	0.000767	2.9183	2.9727	0.0
H2O(a)(Reflector)	0.0	0.0	0.000336	2.9676	2.9865	0.0

5.2.2 Criticality Results for Finite Benchmark Test Cases

In this section we present the results obtained for the criticality calculations corresponding to the finite benchmark test cases whose cross section data is presented in the preceding section.

For all the considered test cases, it can be noted that the calculated k_{eff} approaches 1 as we refine the discretization of each system, which is in accordance with the flat flux approximation. Note that the benchmark cases are critical, then $k_{eff} = 1$ is the exact solution.

The values of k_{eff} obtained as function of the mesh size are presented in Figure 11 for each test case solved. In this figure the mesh size is given by h which is equal to the minimum mesh width used in the discretization of the system. The plots are given with log-log scale in the axis to make it evident the behaviour $k-1 \propto h^{\Lambda}$. Therefore, the value of Λ can be obtained from the gradient of the plots noting that $\log(k-1) = \Lambda \log h$. For all the cases solved Λ tends to 2 when h goes to zero.

The reference benchmark provides the values r^e of the scalar fluxes normalized to the scalar flux at the centre of the system for the positions x corresponding to $x/x_c = 0.25$, 0.5, 0.75 and 1.0, where x_c is the coordinate of the origin of the system. The normalized fluxes we obtained at these points, $\{r_i, i = 1,\ldots,4\}$ where the index i denotes one of the four x/x_c positions, were used to calculate the error $\varepsilon = \frac{1}{4}\sqrt{\sum_{i=1}^{4}(r_i - r_i^e)^2}$ as an indication of the accuracy of our results. These errors as function of the mesh size h are presented in Figure 12. The behaviour $\varepsilon \propto h^{\Lambda}$ is evident again as expected and $\Lambda \approx 0.9$ is obtained when h tends to zero.

Figure 13 shows the CPU time needed to solve each problem as function of the mesh size. It can be seen that the CPU time increases for all the cases when decreasing h, i.e. when increasing the number of meshes in the system. This is expected since a finer discretization leads to a denser collision probability matrix to solve.

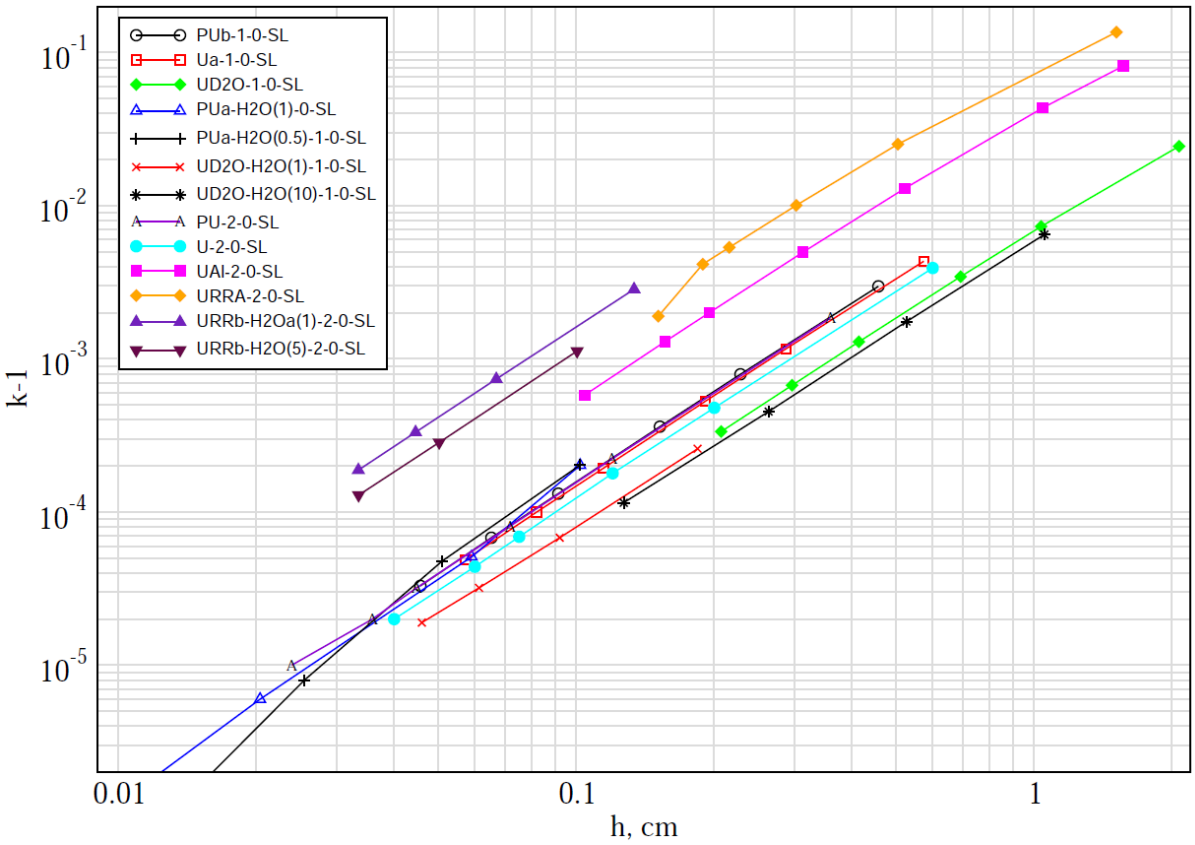


Figure 11: Error in k_{eff} as function of the mesh size

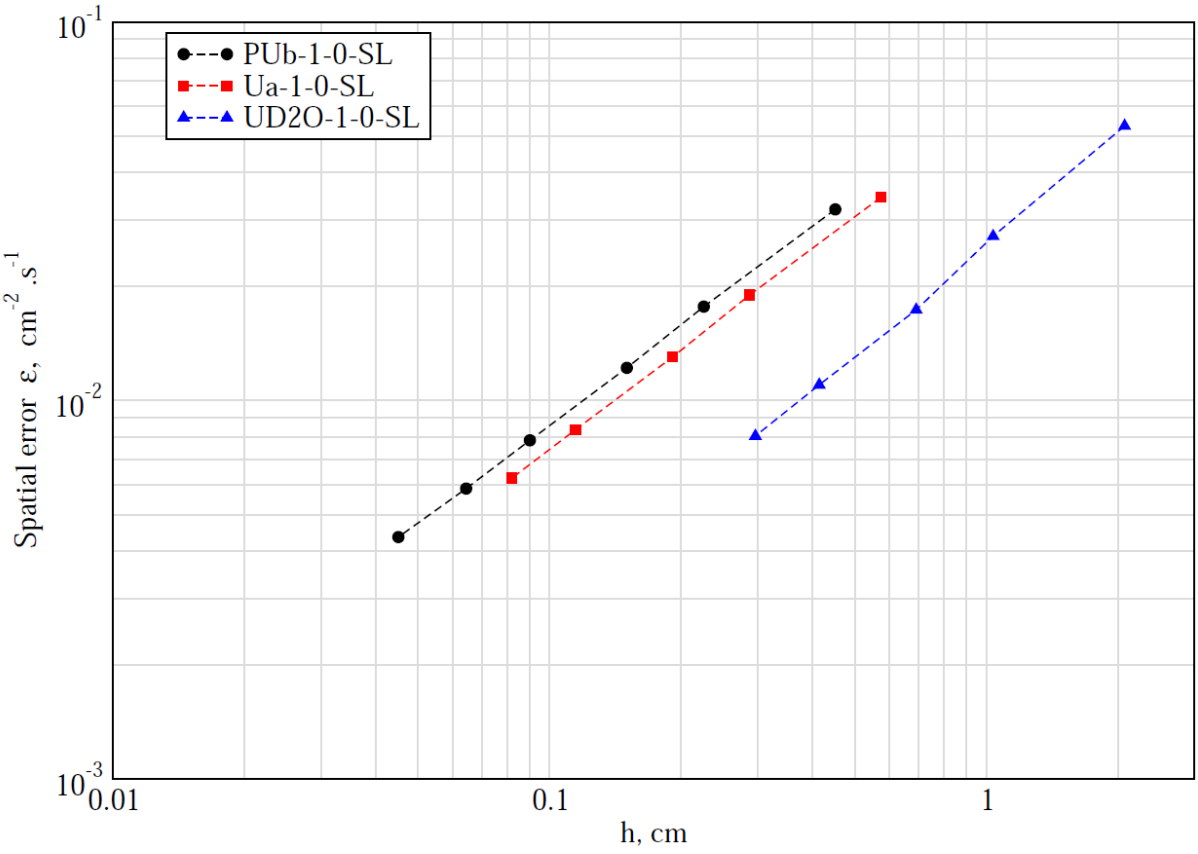


Figure 12: Spatial error ε as function of the mesh size

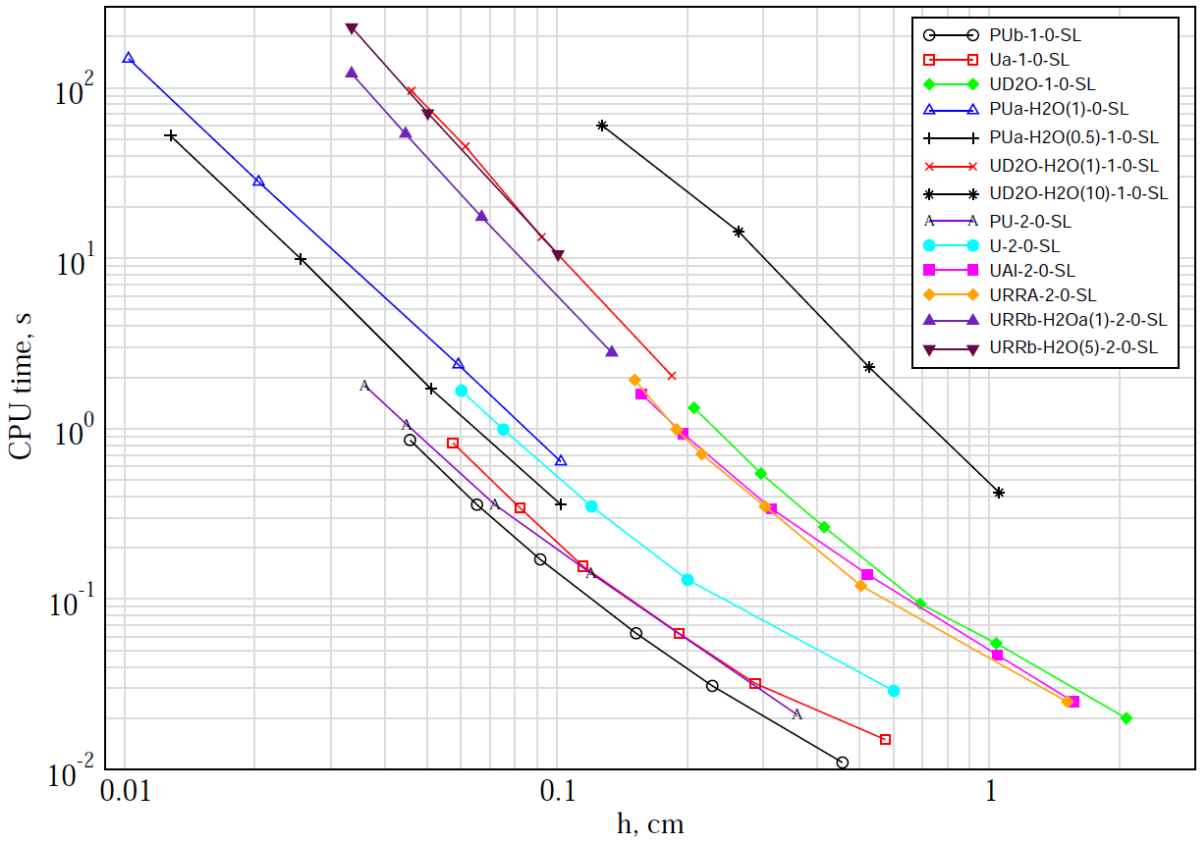


Figure 13: CPU Time as function of the mesh size

Figure 14 shows the shape of the flux obtained with the Oklo code as function of the distance to the core centre for a one medium, two energy group bare system. The expected cosine distribution is obtained in all the other test cases solved for bare cores.

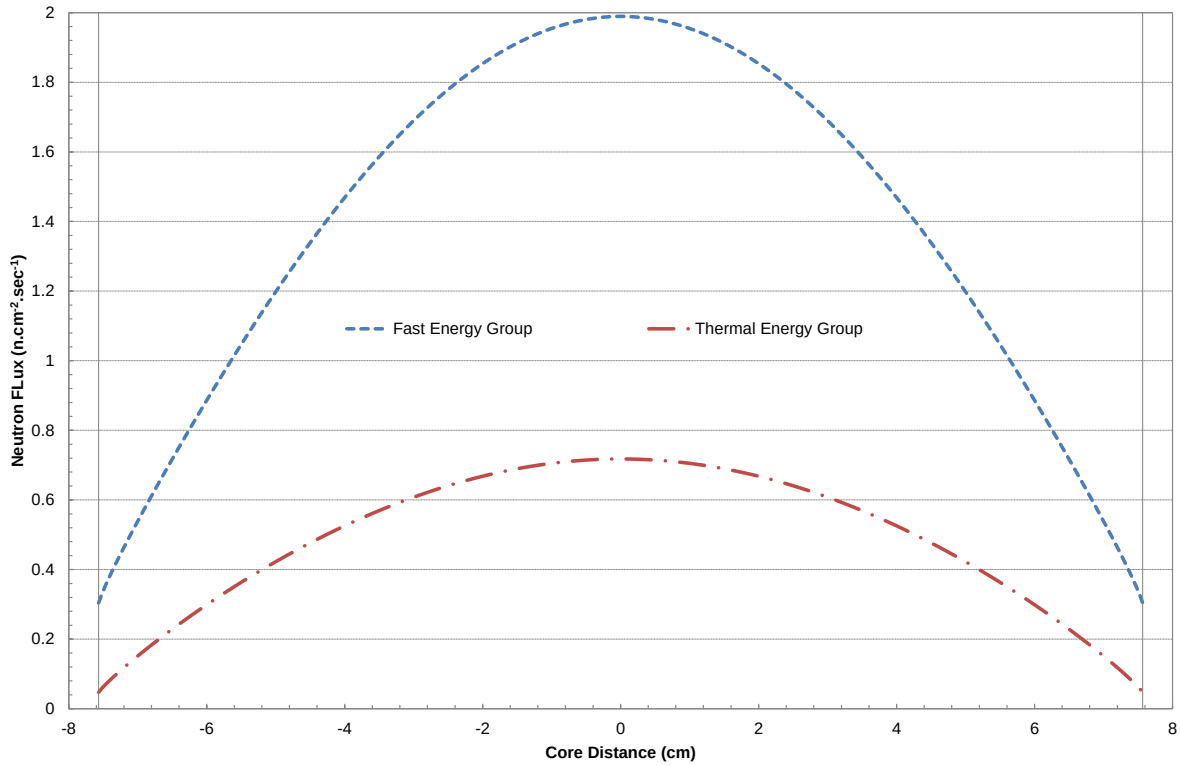


Figure 14: Flux Profile URRa-2-0-SL

For the reflected systems, when the width of the moderator is increased, the required critical dimension of the fuel is decreased significantly compared to the fuel dimension that was required when a smaller width reflector was used. This is due to the reduced neutron leakage with an increased reflector width. There is a limit on the achieved neutron leakage saving due to an increased size of the reflector expressed by the reflector saving concept.

In Figure 15 we show a flux profile for a reflected one medium one energy group case.

For reflected systems with both thermal and fast energy groups, the thermal flux profile increases in the reflector and decreases in the fuel region because the fuel absorbs thermal neutrons whilst the reflector makes fast neutrons thermal by slowing them down. This behaviour can be observed on Figure 16. In this figure we also show the use of reflective boundary condition at the centre of the system and void boundary condition on the right boundary.

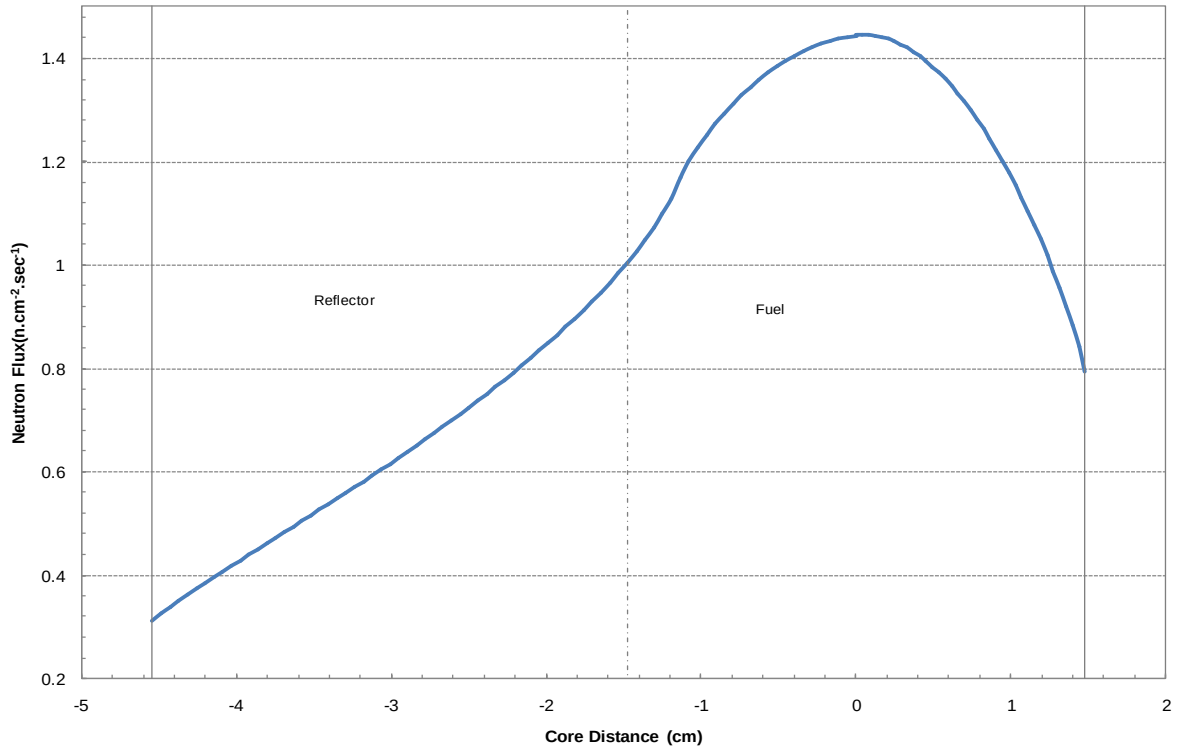


Figure 15: Flux Profile PUa-H2O(1)-1-0-SL

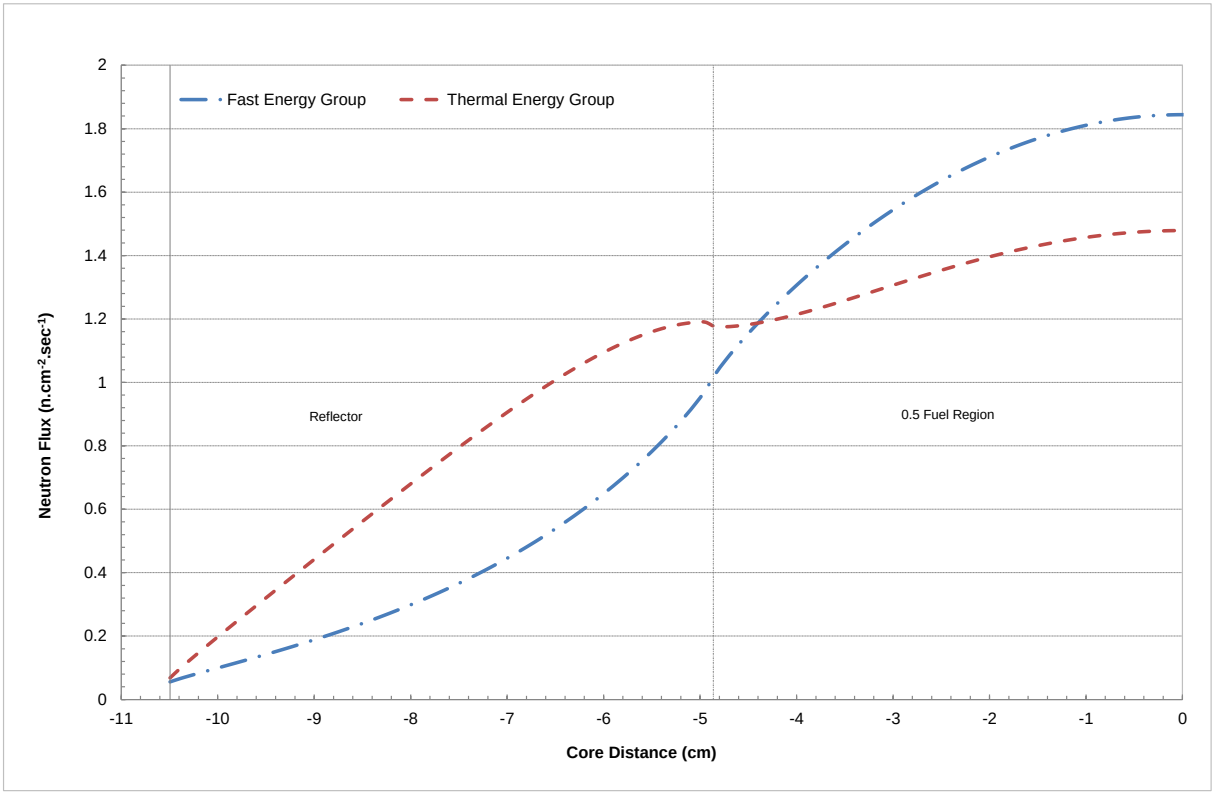


Figure 16: URRb-H2O(5)-2-0-SL Reflected at Fuel Centre

5.2.3 One Medium Six-Energy Group Criticality Results

The test case used for a 6 energy group is not from the benchmark test sets but has typical Material Testing Reactor cross section. The cross section data for the six energy groups is presented in Table 11 and Table 12.

Table 11: Cross Section Data for U-235 6 Energy Group Case

	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
$\nu\Sigma_f$ (cm-1)	1.91E-003	9.55E-004	1.29E-002	2.08E-002	8.78E-002	2.22E-001
χ	7.688380E-01	2.310042E-01	1.581018E-04	0.0	0.0	0.0
Σ_t (cm-1)	1.39E-01	2.83E-01	4.00E-01	4.87E-01	7.82E-01	1.51E+00

Table 12 shows the scattering cross section matrix for this six energy group bare reactor. The unit for the scattering cross sections is cm^{-1}.

Table 12: Scattering Cross Section Matrix for U-235 6 Energy Group Case

$\Sigma_{sg'g}$	1	2	3	4	5	6
1	6.299545E-02	7.47E-002	4.30E-004	2.62E-007	2.79E-008	0.0
2	0.0	1.893247E-01	9.23E-002	5.51E-005	7.92E-006	2.21E-006
3	0.0	0.0	2.749369E-01	9.12E-002	1.29E-002	3.72E-003
4	0.0	0.0	0.0	1.152940E-01	3.00E-001	5.73E-002
5	0.0	0.0	0.0	2.19E-003	3.220751E-01	4.08E-001
6	0.0	0.0	0.0	6.94E-009	4.82E-002	1.341007E+00

There was no critical dimension provided, so the first exercise was to get a critical dimension before observing the flux behaviour on all energy groups. After a few trials a critical dimension was found to be 22 cm.

Criticality results for different discretization schemes with execution time and the total number of outer iterations are shown in Table 13.

Table 13: Keff for Different Discretization for U-235-6-0-SL

Total Number of Meshes	k_{eff}	CPU Time (sec)	No. of outer Iterations
50	0.99712	2.453	13
70	0.999034	4.859	13
95	0.999985	9.797	13

The calculated multiplication factor is approaching 1 as we increase the total number of meshes over which we discretize the domain due to the realization of the flat source approximation. The flux profile for all energy groups over the core distance for this homogenous case is depicted in Figure 17.

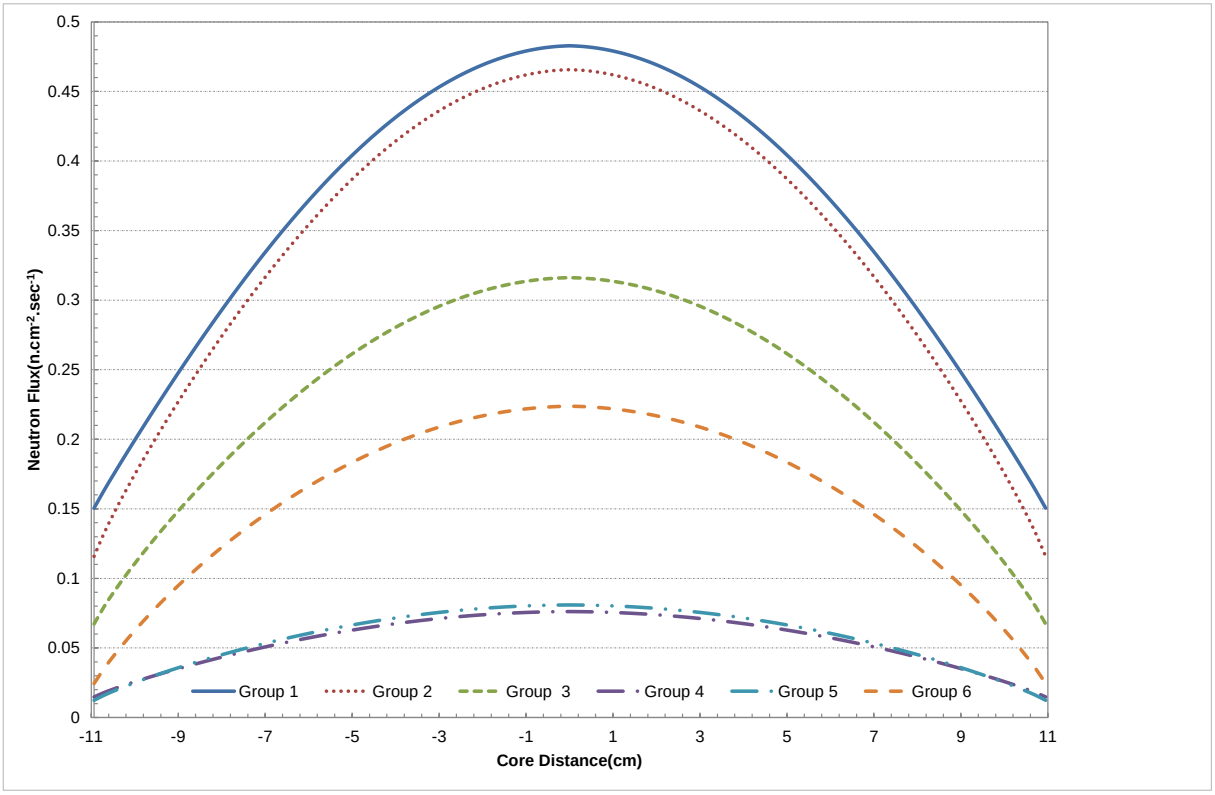


Figure 17: Flux Profile UH2O-6-0-SL

5.2.4 Two Media Six-Energy Group Criticality Results

The test case is similar to the one in the previous Section but with a reflector added on both ends of the slab. The cross section data for the fuel is as provided on Table 11 and Table 12 above for the six energy group homogenous case. The cross section data for the reflector is given in Table 14 and

Table 15.

Table 14: Cross Section Data for H2O Reflector 6 Energy Group Case

	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
Σ_t **(cm^{-1})**	1.610185E-01	3.336887E-01	5.317683E-01	6.515953E-01	1.059497E+00	1.984357E+00

The following table shows the scattering cross section matrix for the reflector. The units for all entries are cm^{-1}.

Table 15: Scattering Cross Section for H2O Reflector 6 Energy Group Case

	1	2	3	4	5	6
1	6.529035E-02	9.619704E-02	5.553896E-04	3.380758E-07	3.608041E-08	0.0
2	0.0	1.999414E-01	1.336196E-01	7.900423E-05	1.135338E-05	3.170939E-06
3	0.0	0.0	3.699083E-01	1.369360E-01	1.901924E-02	5.489903E-03
4	0.0	0.0	0.0	1.751494E-01	3.994077E-01	7.454128E-02
5	0.0	0.0	0.0	1.374267E-03	3.649394E-01	6.866847E-01
6	0.0	0.0	0.0	2.352169E-09	4.144961E-02	1.926983E+00

The critical dimension without a reflector is 22 cm, whilst the critical dimension with a reflector in place is 17.6 cm due to the reflector savings produced by the 10 cm of reflector surrounding the system

The flux profile for the 6 energy group reflected reactor with void boundary condition on the right boundary of the system and reflective boundary condition at the centre of the fuel is shown on Figure 18. Note that the thermal flux increases in the reflector region close to the fuel while faster groups decreases due to slowing down and lack of thermal absorption in the water. The results obtained with void boundary condition on the right boundary of the system and reflective boundary condition at the centre of the fuel are the same as the results obtained with void boundary condition at both ends of the full domain.

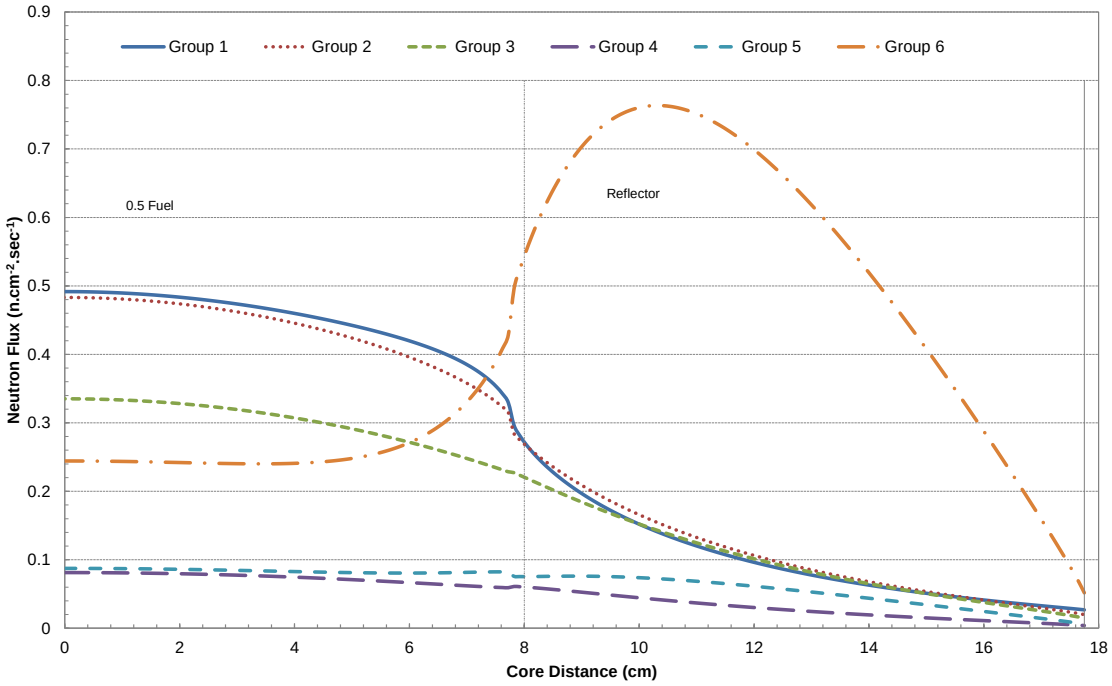


Figure 18: U-H2O-6-0-SL Flux Profile

5.3 Criticality Results Collision Probabilities vs. Discrete Ordinates

This section compares the criticality results obtained using the collision probability code Oklo with criticality results obtained using a discrete ordinates code. Test Case 1 is a bare Pu-239 reactor, Test Case 2 is a reflected one energy group system and Test Case 3 is a reflected 2 energy group system. For all three test cases the collision probabilities code gives a multiplication factor that is closer to 1 than the discrete ordinates code. As we increase the order N of the SN approximation, the results obtained with this method tend to the values obtained with the collision probabilities code. This is due to the exact treatment of the angular variable performed with collision probabilities for isotropic scattering which is approximated by the angular order N of the SN method. The SN code used is a three-dimensional multigroup nodal code which is under development in NECSA. There are no comparisons in this document on the execution time required for both Oklo and the Discrete Ordinates code since the last one is under development and not optimized yet with the standard accelerations of a SN code. The results can serve only as a verification of both codes together with a conceptual analysis of the accuracy of the methods used.

Test Case 1: PUa-1-0-SL

Table 16 shows the cross section data for this system. The system has a critical dimension of 3.707 cm. Table 17 presents the multiplication factors obtained after running the criticality calculations with the collision probabilities code Oklo with different number of meshes of equal size. Table 18 shows the multiplication factors obtained for different SN orders using the discrete ordinates code for 10 and 100 meshes

It can be observed that the SN results tend to the collision probabilities results when the order of the SN approximation increases. The highest order of SN implemented in the Discrete Ordinates code is 16 and it is expected that the accuracy of the result obtained with this code improves using higher SN order.

Table 16: Cross Section Data for Pu-239(a)

ν	$\Sigma_f\ (cm^{-1})$	Σ_s (cm^{-1})	Σ_t (cm^{-1})
3.24	0.081600	0.225216	0.32640

Table 17: Keff Results for Different Discretization using Collision Probabilities Code

Total Number of Meshes	k_{eff}
10	0.997589
100	0.99997
200	0.99999

Table 18: Keff Results for Different SN Orders using Discrete Ordinates Code

	k_{eff}	
SN	m=10	m=100
4	0.94137	0.94182
8	0.98459	0.98518
12	0.99280	0.99342
16	0.99547	0.99610

Test Case 2: PUa-H2O(0.5)-1-0-SL

The cross section data for this system is shown in Table 7. Figure 19 shows the results obtained with the Oklo code for different discretizations of the system. The results shown in Table 20 for different orders of SN correspond to a dense mesh such that the multiplication factors are spatially converged. As in the previous example, it can be seen that when the order of SN increases, the results obtained with the discrete ordinates method tend to the collision probabilities results.

Table 19: Keff Results PUa-H2O(0.5)-1-0-SL Collision Probabilities Code

Total Number of Meshes Per Region	k_{eff}
15,26,15	0.999797
30,52,30	0.999952
60,104,60	0.999992

Table 20: Keff Results PUa-H2O(0.5)-1-0-SL Discrete Ordinates Code

SN Order	k_{eff}
4	0.9539
8	0.99053
16	0.99787

Test Case 3: URRb-H2O(1)-2-0-SL

The cross section data for this system is shown in Table 9 and Table 10. The resulting multiplication factors calculated using the collision probabilities code and the discrete ordinates code are presented in Table 21 and Table 22 respectively. Again, we note that the results from the discrete ordinates code tend to the collision probabilities results as we increase the order of the SN.

Table 21: Keff Results URRb-H2O(1)-2-0-SL Collision Probabilities Code

Total Number of Meshes Per Region	k_{eff}
30,50,30	0.992331
20,200,20	0.99926

Table 22: Keff Results URRb-H2O(1)-2-0-SL Discrete Ordinates Code

SN Order	k_{eff}
2	0.99055
4	0.99921
8	0.99983

5.4 Fixed Source Results

This section presents the results obtained for fixed source problems where neutrons from a time independent source are supplied to a sub-critical system. Test Case 1 presents results for a fixed source problem with a high gradient obtained using the Oklo code compared with results from a discrete ordinates code. Test Case 2 presents results from the two codes for an infinite system.

Test Case 1: High Gradient System

The system is made of two materials with material 1 of size 80 cm surrounded by 2 regions of material 2 of size 40 cm on both ends. The cross section data for both materials making up this system is presented in Table 23.

Table 23: Cross- Section data for High Gradient

Material Name	$\nu\Sigma_f$ (cm^{-1})	Σ_s (cm^{-1})	Σ_t (cm^{-1})	External Source (n.cm-3.sec-1)
Material 1	0.0	0.9	1.0	1.00
Material-2	0.0	0.1	1.0	0.00

The scalar flux profiles for half of this system from the Oklo code and the SN code are plotted in Figure 19. The plots from the two codes show a good agreement even at the interface between the source free material and the material with a source. The discrete ordinates code results approach the collision probabilities results as we increase the order of the SN as shown in Figure 19.

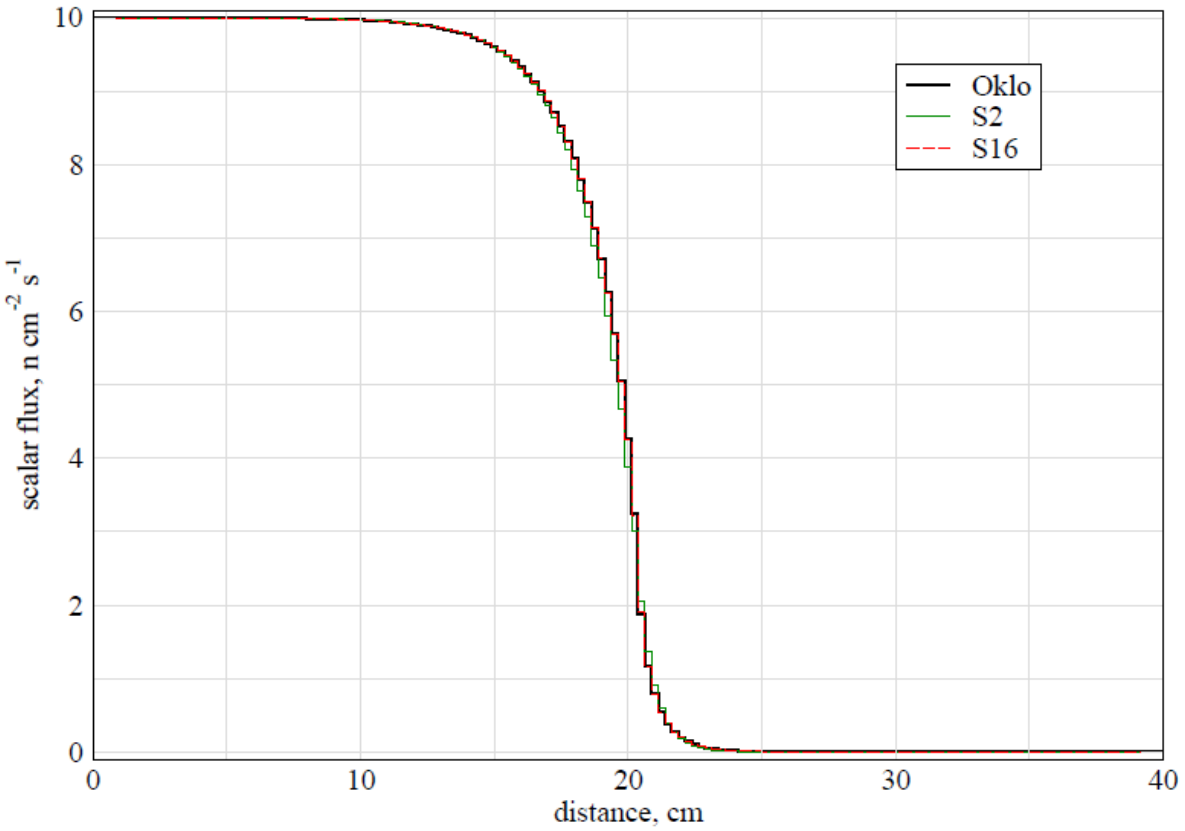


Figure 19: Flux Profiles High Gradient Case SN Results vs. Oklo Results

Test Case 2: Infinite System

This infinite system is made up of a source free region of 0.8 cm surrounded by two regions of 1.4 cm with a source on both sides. The cross section data for the system is shown in Table 24.

Table 24: Cross Section Data Infinite System

Material Name	$\nu\Sigma_f$ (cm^{-1})	Σ_s (cm^{-1})	Σ_t (cm^{-1})	External Source (n.cm-3.sec-1)
Material 1	0.0	2.330	2.35	1.00
Material-2	0.0	0.397	0.717	0.00

The flux profile from the Oklo code for this system is plotted alongside fluxes for different SN orders on the histogram in Figure 20 below. Again we note that there is a good agreement between the fixed source flux profile from the Oklo code and the discrete ordinates code. The discrete ordinates results tend towards the collision probabilities results as we increase the SN order as noted on the criticality calculations.

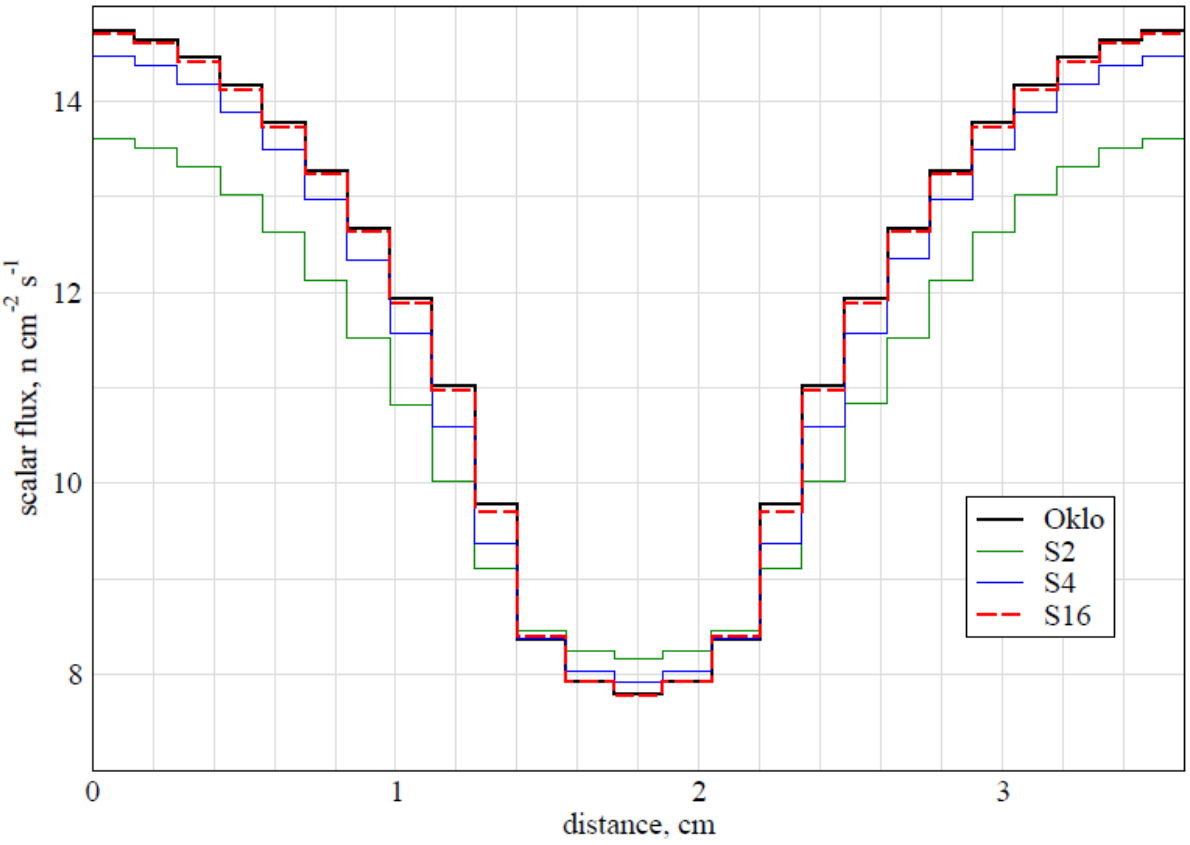


Figure 20: Flux Profile Infinite Fixed Source System Oklo vs. SN Results

5.5 Infinite Homogenous Lattice Results

This section presents the verification results for different sets of test cases from the Los Alamos Benchmark test sets. The results are presented in tabular format showing the actual infinite multiplication factor k_∞ alongside the reference infinite multiplication k_∞ for each test case and the actual and reference flux ratios for test cases with more than one energy group.

5.5.1 One Medium One-Energy Group Results

The test cases Pub-1-0-IN and UD20-1-0-IN are based on the cross sections from Table 6. The cross-section data for the rest of the infinite systems under this category is shown in Table 25.

Table 25: Cross section Data for One Medium One Energy Group Test Cases

Material Name	ν	Σ_f (cm^{-1})	Σ_s (cm^{-1})	Σ_t (cm^{-1})
Pu-239(a)	3.24	0.081600	0.224216	0.32640
U-235(a)	2.70	0.65280	0.248064	0.32640
U-235(b)	2.797101	0.065280	0248064	0.32640
U-235(c)	2.707308	0.065280	0.248064	0.32640
U-235(d)	2.679198	0.065280	0.248064	0.32640
U-235(e)	2.50	0.06922744	0.328042	0.407407

Table 26 shows the results for infinite lattices for one energy group one medium test cases.

Table 26: Calculated Infinite Multiplication Factors vs. Expected Values 1-Energy Group Cases

Test Case Identifier	Reference k_∞ from Benchmarks	Actual k_∞ from Oklo Code	Error (%)
PUa-1-0-IN	2.612903	2.612903	0
Pub-1-0-IN	2.290323	2.290323	0
Ua-1-0-IN	2.250000	2.250000	0
Ub-1-0-IN	2.330917	2.330917	0
Uc-1-0-IN	2.256083	2.256089	2.7E-04
Ud-1-0-IN	2.232667	2.232665	E-05
UD2O-1-0-IN	1.133333	1.133333	0
Ue-1-0-IN	2.1806667	2.1806665	9.1E-06

The calculated values for k_∞ are all in agreement with the reference values from the benchmark test sets at least to 5 decimal places. This shows that the Oklo code can be reliably used to calculate multiplication factors for infinite slabs.

5.5.2 One Medium Two-Energy Group Results

This section presents the results for the infinite multiplication factor k_∞, fluxes for both groups 1 and 2 alongside the reference k_∞ and flux ratios between the two groups from the benchmark test sets. The cross section data for these test cases has already been given Section 5.2.1 of this report. Table 27 shows the result for calculated infinite multiplication factors for two energy group test cases.

Table 27: Calculated Infinite Multiplication Factors vs. Reference Values 2- Energy Group Cases

Test Case Identifier	Reference k_∞ from Benchmarks	Actual k_∞ from Oklo Code
PU-2-0-IN	2.683767	2.683767
U-2-0-IN	2.216349	2.216349
UAL-2-0-IN	2.662437	2.662437
URRa-2-0-IN	1.631452	1.631452
URRb-2-0-IN	1.365821	1.365821
URRc-2-0-IN	1.633380	1.633379
URRd-2-0-IN	1.034970	1.034970
UD2O-2-0-IN	1.000221	1.000221

Table 28 shows the flux for both energy group 1 and energy group 2 and ratios between these fluxes for all the two energy group infinite slab test cases.

Table 28: Calculated Flux Ratios vs. Reference Results 2-Energy Group Cases

Test Case Identifier	Group 1 Flux(n.cm^{-2}sec^{-1})	Group 2 Flux(n.cm^{-2}-sec^{-1})	Reference Flux Ratio from Benchmarks	Actual Flux Ratio from Oklo Code
PU-2-0-IN	1.130745	1.674608	0.675229	0.675229
U-2-0-IN	0.694180	1.461531	0.474967	0.474967
UAL-2-0-IN	3.092190	0.989516	3.124951	3.124951
URRa-2-0-IN	2.112457	0.807914	2.614706	2.614706
URRb-2-0-IN	1.766598	1.505180	1.173679	1.173679
URRc-2-0-IN	1.757613	0.909069	1.933422	1.933422
URRd-2-0-IN	0.129286	0.063897	2.023344	2.023344
UD2O-2-0-IN	0.036727	0.001369	26.822093	26.822093

Both k_∞ and group 2 to group 1 flux ratios shown in Table 27 and Table 28 match the reference values from the benchmark test sets to the last digit of the reference values. These results show that the Oklo code can be used with great confidence to evaluate flux and infinite multiplication factors for infinite lattices. As expected for all test cases the calculated scalar flux is constant per energy group.

6. DISCUSSION OF RESULTS

The code has been proven to be correct using the verification process, which compares the code output against international benchmarks gathered from peer reviewed journals in the nuclear engineering field. The collision probabilities calculated by the developed code Oklo have been checked if they satisfy the reciprocity relation The calculated collision probabilities have also been checked if they satisfy the conditions; that $\sum_j P_{ij} < 1$ for void boundary conditions and $\sum_j P_{ij} = 1$ for infinite systems. The calculated first flight collision probabilities all adhere to these conditions as shown in the results section of the report.

The code developed in the project also calculates k_{eff} that is very close to unit for given critical systems from different materials as demonstrated in the results section. The critical dimensions are reduced for reflected reactors as observed in the reflected reactors compared to the bare reactors due to the reduced neutron leakage in the presence of a reflector.

The calculated k_{eff} approaches 1 with a refinement of the meshes i.e. dividing the domain into infinitesimal meshes results in k_{eff} which is closer to 1. This is due to the assumption of a flat source made in the collision probabilities method. As we refine the mesh we are getting closer to realizing this assumption. However, as we increase the meshes per domain of the problem the execution time increases, because this has a direct influence on the size of the resulting matrix that is solved for collision probabilities. It has been shown the behaviour of k_{eff} approaching 1.0 for critical benchmark test cases and the increasing execution time with a refinement in discretization respectively.

The code results for calculating k_{∞} and average scalar fluxes per group for infinite systems matched the values given in the benchmark test sets to at least the fifth decimal place. The infinite lattice calculations required more execution time because of the increased computation added for the first flight collision probabilities by calculating them over the range $-\infty$ to $+\infty$.

The criticality results from the collision probabilities code give in general values closer to 1.0 compared to results from a discrete ordinates code using S16 due to the exact treatment of the angular variable by the collision probabilities method, which is approximated by the angular order of N in the SN. With an increase in the SN order the results from the discrete ordinates code tend towards the collision probabilities code result.

The fixed source results from the Oklo code show a good agreement with results from the discrete ordinates code. Again, as noted for the criticality results, the discrete ordinates results move towards the collision probabilities results as we increase the SN order. The disadvantage of the collision probability method is the requirement for sizeable computer memory to store the first flight collision probability matrices that can become very dense, the more mesh points we consider under a domain. The size of the matrix is given by N^2 if N is the number of equivalent meshes the domain is divided into.

7. CONCLUSION

The project has satisfied the objectives that were set at the start i.e. developing a code that solves the integral transport equation using the method of Collision Probabilities for both multiplicative and non-multiplicative systems. The theory section of the report gave an introduction of the integro-differential transport equation alongside the integral transport equation. The integral transport equation was then transformed to a multigroup form, eliminating the energy variable in the equation leaving the equation only dependent on space. The method of Collision Probabilities and its use in solving the integral transport equation was then discussed, culminating with a system of equations for which the code had to be designed. The multigroup scheme was depicted in a flow diagram with the equations that were already transformed to a form that can be programmed.

The solution was implemented using C++, on the Microsoft Visual Studio 2005 platform. The system of equations was solved using a LU decomposition library from GNU Scientific Library. Test cases for checking the calculated Collision Probabilities were then presented, and the results were shown to be correct.

The test cases for both criticality and fixed source calculations matched the reference results from the benchmarks. The results from the code were also compared with results from a discrete ordinates code for both criticality and fixed source calculations, and the discrete ordinates results approached the Oklo results as we increase the order N, which is due to the exact treatment of the of the angular variable performed with collision probabilities for isotropic scattering. As a consequence of this positive outcome from the verification of the results from the code developed in the project, the code can be used with confidence to perform lattice calculations.

The project may be improved in the future by implementing the reciprocity relation and compare the performance with the current implementation.

8. REFERENCES

[1] World Nuclear Association: www.world-nuclear.org.

[2] Lewis E.E. and Miller W.F., Computational Methods of Neutron Transport, 1984, John Wiley & Sons Inc.

[3] Stammler R.J.J, Methods of Steady State Reactor Physics in Nuclear Design, Academic Press, 1983.

[4] Cacuci D.A., Handbook of Nuclear Engineering Volume 1- Volume 4, Springer Reference, 2010.

[5] Zamonsky O.M. and Gho C.J., Notes of the course on Neutron Physics corresponding to the Nuclear Engineering career at Instituto Balseiro, National University of Cuyo, Argentina (in Spanish).

[6] Aragones J.M. et al, Status and limits of current methods for plant analysis, Nuclear Reactor Integrated Simulation Project (NURISP), 2010.

[7] Benoist P. Lectures on Neutron Transport Theory, French Atomic Energy Commission, 1986.

[8] GNU Scientific Library: www.gnu.org/software/gsl/.

[9] Sood A. et al, Analytical Benchmark Test Set for Criticality Code Verification, Progress in Nuclear Engineering, Vol. 42, No.1, pp 55- 106,2003, Elsevier Science Limited.

9. ADDENDUM

9.1 Setting Up Input File

The input file must be of comma separated values type.

Table 29: Input File Fields

Identifier	Description
Oklo Input Settings File,	Optional Description of File
<GenericSettings>,	Marks the start of setting for the system under consideration
<NGROUPS>,	Total number of energy groups
<NCELL>,	Total number of cells making up system
<NMAT>,	Gives the total number of distinct materials system is made of.
<MATMAP>,	Identifies what type of material each cell is made of
<NUMMESHES>,	Gives the total number of meshes each material is discretized into
<Epsilon_Keff>	Epsilon for convergence for multigroup calculations
=======Material 1: PU-239========,	Marks the start of properties of named material
<DELTA>,	Size for material
<CHI>,	χ for all energy groups under this key
<XT>,	Total Cross section
<XS>,	Marks the start of the scattering cross section matrix
<NUF>,	A product of ν and Σ_f
<SOURCE>,	Marks the start of the external source, applicable for Fixed source problems only.
=====END FILE====,	Marks the end of the file

This is an example of a one medium one energy group input file:

Oklo Input Settings File,

<GenericSettings>,

<NGROUPS>,

1,

<NCELL>,

1,

<NMAT>,

1,

<MATMAP>,

1,

<NUMMESHES>,

4,

<Epsilon_Keff>,

0.000001,

=======Material 1: PU-239========,

<DELTA>,

3.707444,

<CHI>,

1,

<XT>,

0.32640,

<XS>,

0.225216,

<NUF>,

0.264384,

<SOURCE>,

0.0,

=====END FILE====,

The following is an example of a two energy group one medium system input file.

Oklo Input Settings File,

<GenericSettings>,

<NGROUPS>,

2,

<NCELL>,

1,

<NMAT>,

1,

<MATMAP>,

1,

<NUMMESHES>,

500,

<Epsilon_Keff>,

0.000001,

======Material 1: URR============,

<DELTA>,

15.133706 ,

<CHI>,

1.0, 0.0

<XT>,

0.65696, 2.52025

<XS>,

0.62568,0.029227

0.0,2.44383

<NUF>,

0.002621,0.12658

<SOURCE>,

0.0,0.0

========END FILE=================,

The following is a one energy group two media system input file:

Oklo Input Settings File,

<GenericSettings>,

<NGROUPS>,

1,

<NCELL>,

3,

<NMAT>,

2,

<MATMAP>,

2,1,2

<NUMMESHES>,

144,128,144

<Epsilon_Keff>,

0.000001,

======Material 1: UD2O============,

<DELTA>,

16.856192,

<CHI>,

1,

<XT>,

0.54628,

<XS>,

0.464338,

<NUF>,

0.0928676,

<SOURCE>,

0.0,

======Material 2: H2O============,

<DELTA>,

18.30563 ,

<CHI>,

0,

XT>,

0.54628,

<XS>,

0.491652,

<NUF>,

0.0,

<SOURCE>,

0.0,

=======END FILE==================,

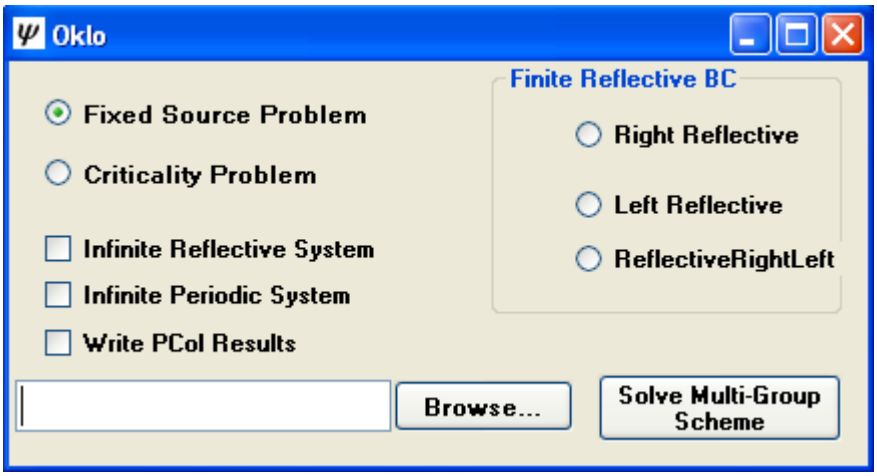
9.2 Running Criticality Calculations

To run criticality calculations for s system follow the steps:

i. Set up input file as described in Section 9.1 of this appendix

ii. Run Oklo executable

iii. Open input file for system being modelled using the browse button



iv. Check Criticality Problem radio button

v. Check Write PCol Results if you want to get output file for collision probabilities

vi. If this is an infinite system, check the infinite system check box.

vii. Click Solve Multigroup Scheme button

viii. When completed with criticality calculation, the system will display the message “finished running case”

ix. Click OK

x. View output file on executable folder

9.3 Running Fixed Source Calculations

To run fixed source calculations for s system follow the steps:

i. Set up input file as described in Section 9.1 of this appendix

ii. Note that you need to specify the external source under <Source>, in input file for Fixed Source problems

iii. Run Oklo executable

iv. Open input file for system being modelled using the browse button

v. Check Fixed Source Problem radio button

vi. If you are modelling an infinite system, check the infinite system checkbox.

vii. Check Write PCol Results if you want to get output file for collision probabilities

viii. Click Solve Multigroup Scheme button

ix. When done with Fixed Source calculation system will display the message “finished running case”

9.3.1 Running Infinite Lattice Calculations

To run infinite lattice calculations for system follow the steps:

i. Set up input file as described in Section 9.1 of this appendix

ii. Run Oklo executable

iii. Open input file for system being modelled using the browse button

iv. Check Criticality problem radio button

v. Check infinite system checkbox.

vi. Check Write PCol Results if you want to get output file for collision probabilities

vii. Click Solve Multigroup Scheme button

viii. When done with Infinite System calculation system will display the message “finished running case”

ix. Note that the infinite system calculations will take a longer period to complete due to the calculation of first flight collision probabilities over the range $-\infty$ and $+\infty$

9.4 Output File Content and Interpretation

The output file contains the following Section

- Material Properties, contains the cross section data for materials making up the system and generic modelling information like convergence epsilon
- CPU time for calculations: contains the time in seconds the system took to execute the test case for the system under consideration
- Keff per Generation or Kinf per Generation which gives a list of effective multiplication faction or infinite multiplication factors for all iterations depending on whether the problem being modelled is a criticality calculation or an infinite system.
- Fluxes Per Mesh Per Generation Per Energy Group: Contains Matrices for average scalar neutron flux per mesh per energy group for all generations. Each column represents flux for a particular energy group starting with the fast energy group to the thermal energy group. Each row per represents the average scalar per mesh index corresponding to the count of that row

9.5 Program Functions and Descriptions

The following table contains a list of the main functions contained in the code and a brief description of what each function does.

Table 30: Code Functions and Descriptions

Function Name	Description
collisionProbs.cpp	Calculates collision probability
Container.cpp	Defines the data structure into which we read input file. i.e. cross section data for system and generic data for system being modeled
cPsInfiniteLattice.cpp	Calculates first flight collision probabilities for an infinite system
cPsPeriodicLattice	Calculates first flight collision probabilities for a periodic system
cPsReflectiveLattice	Calculates first flight collision probabilities for a reflective system
expInt.cpp	Calculates exponential integrals
fissionSource.cpp	Calculates the fission source
fixedSourceFluxCalc.cpp	The main function for fixed source calculations.
guessFlux.cpp	Calculates the guess flux at the start of the multigroup iteration
matrixSolver.cpp	Solves the resulting system of linear equations using LU decompositions for average scalar flux.
multiGroupFluxCalc.cpp	The main function for criticality calculations
multiGroupSource.cpp	Calculates the source by combining scattering and fission sources for criticality calculations
multiplicationFactorCalc.cpp	Evaluates the multiplication factor for criticality calculations or infinite multiplication factor for infinite systems
scatteringSource.cpp	Calculates the scattering source into the energy group under consideration
sourceExt.cpp	Calculates the source into a particular group from energy groups different of this group
sourceFixed.cpp	Calculates source for fixed source problems
tau.cpp	Calculates tau for use in calculating first flight collision probabilities
writeCollision.cpp	Writes calculated first flight collision to file
writeOutput.cpp	Writes results for system being modeled for all three problem types.