MINISTRY OF EDUCATION
FEDERAL UNIVERSITY OF RIO GRANDE DO SUL
GRADUATE PROGRAM IN MECHANICAL ENGINEERING

# MONTE CARLO SIMULATOR PROJECT FOR NEUTRON TRANSPORT WITH CONTINUOUS ENERGY: SHIELDING, CRITICALITY AND SPECTRAL ANGULAR NEUTRON FLUX ANALYSIS

by

Luiz Felipe Fracasso Chaves Barcellos

A Thesis for the degree of
Doctor in Engineering

Porto Alegre, March $15^{th}$, 2021

MONTE CARLO SIMULATOR PROJECT FOR NEUTRON TRANSPORT WITH CONTINUOUS ENERGY: SHIELDING, CRITICALITY AND SPECTRAL ANGULAR NEUTRON FLUX ANALYSIS

by

Luiz Felipe Fracasso Chaves Barcellos
Mestre

A Thesis submitted to the committee of the Graduate Program in Mechanical Engineering (PROMEC), from the Engineering School of the Federal University of Rio Grande do Sul (UFRGS), as part of the necessary requirements to obtain the degree of

Doctor in Engineering

Field of Study: Fenômenos de Transporte

Advisor: Prof. Dr. Bardo Ernst Josef Bodmann

Approved by:

Prof. Dr. Antônio Carlos Marques Alvim .....COOPE / UFRJ

Prof. Dr. Celso Marcelo Franklin Lapa .....IEN / CNEN

Prof. Dr. Daniel Artur Pinheiro Palma .....PPGIEN / CNEN

Prof. Dr. Rogério José Marczak .....PROMEC / UFRGS

Prof. Dr. Fernando Marcelo Pereira
Coordinator of PROMEC

Porto Alegre, March $15^{th}$, 2021

Dedico esta tese a meus pais, Henrique e Rosana, por todo o amor, carinho e apoio ao longo dos anos.

*Alea iacta est*

Gaius Julius Caesar

## ACKNOWLEDGEMENTS

I thank my professors Bardo Ernst Josef Bodmann and Marco Túllio Menna Barreto de Vilhena, not only for their teachings and counseling, but also for their friendship.

I am grateful to my friend and colleague Daniel Gustavo Benvenutti for his help with the finalization of this work.

I also thank all friends and family for all the companionship and support, specially Henrique Macedo de Azevedo and Ana Paula Ost.

This author also thanks CAPES for the financial support.

**RESUMO**

Esta tese relata o projeto desenvolvido durante o doutorado do autor. Isso engloba o desenvolvimento de um software de simulação de Monte Carlo de licença aberta para o transporte de nêutrons em materiais de núcleo de reatores. Esta ferramenta consiste em um programa C ++ seguindo os paradigmas recentes em computação de alto desempenho, e leva em consideração as dimensões físicas contínuas (posição, direção de movimento e energia) do espaço de fase do transporte de nêutrons. Dois modelos de espalhamento são implementados e discutidos. Estes são baseados na hipótese de alvo em repouso e no modelo de gás livre, e a diferença entre os modelos é apresentada. Além disso, o fluxo de nêutrons angular espectral simulado é testado como uma solução da equação de transporte de Boltzmann nas suas sete dimensões. Neste trabalho, um cenário de blindagem, um benchmark de criticidade do livro International Handbook of Evaluated Criticality Safety Benchmark Experiments e duas simulações em meio multiplicativo, cada uma com um modelo de espalhamento diferente, foram simulados para apresentar as capacidades do software. O simulador mostra-se versátil nos diferentes tipos de resultados que podem ser obtidos, e.g. fluxo de nêutrons, densidade de nêutrons, taxas de reação, criticidade, entre outros. Por último, os fluxos espectrais de nêutrons são parametrizados (na faixa de $[10^{-14},\ 10^{1}]MeV$) e essas funções são verificadas como possíveis soluções da equação de transporte de Boltzmann.

Palavras-chave: Transporte de Nêutrons; Método Monte Carlo; Modelos de Espalhamento; Fluxo Angular Espectral de Nêutrons; Equação de Transporte de Boltzmann.

**ABSTRACT**

This thesis reports on the project developed during the author's doctorate. This encompasses the development of an open license Monte Carlo simulation software for neutron transport in reactor core materials. This tool consists of a C++ program following recent paradigms of advanced power computing, and takes into account the continuous physical dimensions (position, direction of motion and energy) of the neutron transport phase space. Two scattering models are implemented and discussed. These are based on the target at rest hypothesis and the free gas model, and the difference between models is shown. Also, the simulated spectral angular neutron flux is tested as a solution of the seven-dimensional Boltzmann transport equation. In this work a shielding scenario, a criticality benchmark from the International Handbook of Evaluated Criticality Safety Benchmark Experiments book, and two simulations in a multiplicative medium, each with a different scattering model, were simulated in order to portray the software capabilities. The simulator shows itself versatile in the different kind of results that can be obtained, e.g. neutron flux, neutron density, reaction rates, criticality, among others. Lastly the spectral neutron fluxes are parametrized (in the range of $[10^{-14},\ 10^{1}]MeV$) and these functions are verified as possible solutions of the Boltzmann transport equation.

Keywords: Neutron Transport; Monte Carlo Method; Scattering Models; Spectral Angular Neutron Flux; Boltzmann Transport Equation.

## CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ACRONYMS AND ABBREVIATIONS

| | |
|---|---|
| ENDF | Evaluated Nuclear (reaction) Data File |
| MCNP | Monte Carlo N-Particle Transport Code |

## LIST OF SYMBOLS

### Latin Symbols

| | |
|---|---|
| $A$ | Atomic mass, $a.m.u$ |
| $dx$ | Differential thickness, $m$ |
| $e$ | Enrichment of uranium dioxide |
| $E$ | Energy, $MeV$ |
| $f$ | Fission reaction |
| ${}_2^4He^{+2}$ | Alpha particle |
| $I$ | Mono-energetic neutron beam, $^1/_{m^2\,s}$ |
| $k_B$ | Boltzmann constant, $8.6173 \times 10^{11}\ MeV/K$ |
| $L$ | Total length traveled by the neutron, $cm$ |
| $m_k$ | Number of neutrons emitted by reaction "k" |
| $M$ | Molecular weigh, $^g/_{mol}$ |
| $n$ | neutron in nuclear reaction schemes |
| $n(\vec{r}, \vec{\Omega}, E, t)$ | Neutron density, $^1/_{cm^3}$ |
| $N$ | Atomic density, $^1/_{m^3}$ |
| $N_{235}$ | Atomic density of U-235, $^1/_{m^3}$ |
| $N_{238}$ | Atomic density of U-238, $^1/_{m^3}$ |
| $N_A$ | Avogrado's number, $6.0221 \times 10^{23} {}^1/_{mol}$ |
| $\vec{r}$ | Position vector in the Cartesian system |
| $p$ | Proton |
| $p_i$ | Percentage of $S$ travelled in region "i" |
| $Q$ | Excitation energy of target nucleus, $MeV$ |
| $r$ | Weight percentage |
| $R$ | Reaction rate per square meter, $^1/_{m^2\,s}$ |
| $S$ | Multiple of mean free path |
| $t$ | time, $s$ |
| $T$ | temperature, $K$ |
| $v$ | Absolute value of the velocity vector, $^m/_s$ |
| $v_{CM}$ | Center of mass speed, $^m/_s$ |
| $v_{CM,1}$ | Neutron speed in the center of mass system before the collision, $^m/_s$ |

| | |
|---|---|
| $v'_{CM,1}$ | Neutron speed in the center of mass system after the collision, $^{m}/_{s}$ |
| $v_{CM,2}$ | Nucleus speed in the center of mass system before the collision, $^{m}/_{s}$ |
| $v'_{CM,2}$ | Nucleus speed in the center of mass system after the collision, $^{m}/_{s}$ |
| $v_{Lab,1}$ | Neutron speed in the laboratory system before the collision, $^{m}/_{s}$ |
| $v'_{Lab,1}$ | Neutron speed in the laboratory system after the collision, $^{m}/_{s}$ |
| $v_{Lab,2}$ | Nucleus speed in the laboratory system before the collision, $^{m}/_{s}$ |
| $v'_{Lab,2}$ | Nucleus speed in the laboratory system after the collision, $^{m}/_{s}$ |

**Greek Symbols**

| | |
|---|---|
| $\alpha$ | Angle in the spherical coordinate system ($radian$) |
| $\beta$ | Angle in the spherical coordinate system ($radian$) |
| $\gamma$ | Gamma radiation |
| $\theta$ | Angle in the spherical coordinate system ($radian$) |
| $\lambda$ | Mean free path, $m$ |
| $\nu$ | Average fission neutrons |
| $\rho$ | Density, $^{kg}/_{m^3}$ |
| $\sigma$ | Microscopic cross section, $barn$ |
| $\sigma_a$ | Microscopic absorption cross section, $barn$ |
| $\sigma_s$ | Microscopic scattering cross section, $barn$ |
| $\sigma_t$ | Microscopic total cross section, $barn$ |
| $\Sigma_k$ | Macroscopic cross section of reaction "k", $cm^{-1}$ |
| $\Sigma_t$ | Total macroscopic cross section, $cm^{-1}$ |
| $\phi$ | Neutron flux, $^{1}/_{cm^2\,s}$ |
| $\Phi$ | Angle in the spherical coordinate system ($radian$) |
| $\chi_f(E)$ | Prompt fission spectrum |
| $\psi$ | Angle in the spherical coordinate system ($radian$) |
| $\vec{\Omega}$ | Direction vector in the spherical coordinate system |
| $\vec{\Omega}'$ | Direction vector in the spherical system |
| $\vec{\Omega}_f$ | Final neutron direction vector |
| $\vec{\Omega}^*_f$ | Generalized final direction vector |
| $\vec{\Omega}_i$ | Initial neutron direction vector |
| $\vec{\Omega}_P$ | Auxiliary direction vector |
| $\vec{\Omega}_Q$ | Auxiliary direction vector |

# 1 INTRODUCTION

Nuclear reactor analysis is dependent on the knowledge of the spectral angular neutron flux, given by the solution of the Boltzmann transport equation. This is an equation defined on a seven-dimensional phase space (space, time, kinetic energy and solid angle) and until today the determination of a solution is still a challenge. This equation describes transport of neutral particles in the absence of electromagnetic fields, i.e. it does not take into account any electrical charge of particles nor their magnetic moments (spin).

One possible solving technique for such a problem is the Monte Carlo method. One advantage of this approach is that it allows to solve the Boltzmann equation in all seven dimensions by sampling in a continuous phase space. Sometimes, the information about a given state of a single system is unknown (or not of interest), but, nevertheless, the state of an ensemble of systems can be defined by probability concepts [Reif, 2009; Tolman, 1979]. This is a great advantage when using the Monte Carlo method. Nuclear interaction, e.g. fission, have intermediary states that are not yet well described, and even if they were, the definition of these states would not be relevant for the majority of simulations, but their resulting distributions are. Thus, when simulating a process for great number of particles (systems), expectation values can be found from its ensemble.

The Monte Carlo method has a convergence characteristics that is proportional to $N^{-1/2}$, in which $N$ is the number of simulated particles and this uncertainty is independent of the number of dimensions, whereas numerical methods, such as diffusion theory based and other discrete methods, result not only in the reduction of the seven-dimensional phase space but also have an uncertainty that scales according to $M^{-k/d}$, in which $d$ is the number of dimensions and $M$ the number of discretization points [de Camargo, 2011].

The Monte Carlo simulator, which with its actual state is reported in this work, began its development with Dayana de Carmargo in de Camargo, 2011, and had subsequent advancements in de Camargo et al., 2013, Barcellos et al., 2015, Chaves Barcellos, 2016, Barcellos et al., 2017a, 2021. Some of the simulator features are presented in Barcellos et al., 2017b, 2019, 2020, Chaves Barcellos et al., 2021. This work presents the latest and completely revised and extended version of the program. The present code was written as a multithread C++ program following recent paradigms of advanced power computing,

and takes into account the continuous physical dimensions (position, direction of motion and energy) of the neutron transport phase space.

The simulator does not use integrals over energy invervals, i.e. energy groups, but is able to classify neutrons as part of probability distributions, in the present version one for thermal neutrons, one for fission born neutrons, and another for neutrons in the slowing down process. The first two distributions have known shape, but unknown population, and the third *a priori* unknown shape and population. The program also interpolates cross section data in order to crate compiled continuous functions.

This version may be included as a patch in GEANT [Agostinelli et al., 2003; Collaboration, 2020b,a], and provide the simulation part that involves neutrons, while interactions from ionizing particles are provided by the existent implementation of the huge GEANT library. Differently from other Monte Carlo simulators, such as MCNP [Team, 2003], the main objective of the present software is to find a function for the neutron flux, and to assess whether the function is a candidate solution of the Boltzmann transport equation.

One particular feature of the present implementation, in form of a physical Monte Carlo, is that it is not designed for specific problems only, but allows to use the created simulation data set for a variety of analysis from the same data. Thus the spectral angular neutron flux, obtained from parametric inference, may be evaluated as a candidate for a solution of the Boltzmann transport equation. More specifically, in this work, the spectral neutron flux is parametrized in the range $[10^{-14}, 10^{1}]\, MeV$ and verified whether it solves to a certain approximation a simplified version of the Boltzmann equation. This is a first step towards a solution in the whole parametrized phase space.

## 2 TRANSPORT EQUATION

The *neutron transport equation*, that is also know as *Boltzmann equation*, is the equation that describes the behavior of the angular neutron flux, for instance, in a reactor. Let $\phi(\vec{r}, \vec{\Omega}, E, t)$ be the angular neutron flux, defined in Equation 2.1 as the angular neutron density times the speed of the neutrons.

$$\phi(\vec{r}, \vec{\Omega}, E, t) \equiv v\, n(\vec{r}, \vec{\Omega}, E, t) \tag{2.1}$$

in which $v$ is the absolute value of the velocity vector of the neutron and $n(\vec{r}, \vec{\Omega}, E, t)$ is the angular neutron density, i.e. the density of neutrons at position $\vec{r}$ that travel in the direction $\vec{\Omega}$ (within the solid angle element $d\vec{\Omega}$) with kinetic energy $E$ (in the range $[E,\ E + dE]$) at time $t$.

It is not the objective of this work to review the whole derivation of the transport equation, this is already very well documented in several sources, such as Duderstadt and Martin, 1979. The most important is the interpretation of the transport equation and the knowledge of the meaning of each of its terms. The Boltzmann equation, as it appears in Sekimoto, 2007, is given by Equation 2.2.

$$\frac{1}{v}\frac{\partial}{\partial t}\phi(\vec{r}, \vec{\Omega}, E, t) + \vec{\Omega} \cdot \nabla\phi(\vec{r}, \vec{\Omega}, E, t) + \Sigma_t(\vec{r}, E, t)\phi(\vec{r}, \vec{\Omega}, E, t) = q(\vec{r}, \vec{\Omega}, E, t) \tag{2.2}$$

in which $\Sigma_t$ is the total macroscopic cross section and $q(\vec{r}, \vec{\Omega}, E, t)$ represents a neutron source term given by Equation 2.3.

$$q(\vec{r}, \vec{\Omega}, E, t) = \int_0^\infty dE' \int_{4\pi} d\vec{\Omega}'\Sigma(\vec{r}, E' \to E, \vec{\Omega}' \to \vec{\Omega})\phi(\vec{r}, \vec{\Omega}', E', t) + S(\vec{r}, \vec{\Omega}, E, t) \tag{2.3}$$

in which $S(\vec{r}, \vec{\Omega}, E, t)$ is called the external neutron source, i.e. it is a neutron source independent of the angular neutron flux. The term $\Sigma(\vec{r}, E' \to E, \vec{\Omega}' \to \vec{\Omega})$ includes all neutron emission reactions, such as scattering, fission, and $(n, 2n)$ reactions. It can then be rewritten as presented in Equation 2.4.

$$\Sigma(\vec{r}, E' \to E, \vec{\Omega}' \cdot \vec{\Omega}) = \sum_k m_k(E')\Sigma_k(\vec{r}, E')p_k(E' \to E, \vec{\Omega}' \cdot \vec{\Omega}) \tag{2.4}$$

in which $\Sigma(\vec{r}, E' \to E, \vec{\Omega}' \to \vec{\Omega})$ was considered as a function of $\mu_0 \equiv \vec{\Omega}' \cdot \vec{\Omega}$, instead of the two variables $\vec{\Omega}'$ and $\vec{\Omega}$, this is due to cylindrical symmetry along the axis aligned with the incoming particle's direction. Also in Equation 2.4 $m_k(E')$ represents the number of secondary neutrons released by reaction $k$, e.g 1 for scattering, 2 for $(n, 2n)$ reactions, and $\nu$ for fission. $\Sigma_k(\vec{r}, E')$ is the macroscopic cross section of interaction $k$. Finally the term $p_k(E' \to E, \vec{\Omega}' \cdot \vec{\Omega}) dE d\vec{\Omega}$ represents the probability density that reaction $k$ will release a neutron of energy $E$ in the range $[E,\ E + dE]$ and direction $\vec{\Omega}$ in the infinitesimal solid angle $d\vec{\Omega}$. In the case of fission $p_f(E' \to E, \vec{\Omega}' \cdot \vec{\Omega})$ is given by Equation 2.5.

$$p_f(E' \to E, \vec{\Omega}' \cdot \vec{\Omega}) = \frac{1}{4\pi} \chi_f(E) \tag{2.5}$$

in which $\chi_f(E)$ is given by Equation 3.12. In the case of scattering $p_s(E' \to E, \vec{\Omega}' \to \vec{\Omega}, A)$ depends on more details, since it depends also on the mass number $A$ of the target nucleus, and $\vec{\Omega}'$ is not independent of $\vec{\Omega}$, as will be shown on Chapter 4.

The remaining terms of Equation 2.2 can be labelled as in Equation 2.6, in order to facilitate their description.

$$\underbrace{\frac{1}{v}\frac{\partial}{\partial t}\phi(\vec{r}, \vec{\Omega}, E, t)}_{\text{I}} + \underbrace{\vec{\Omega} \cdot \nabla\phi(\vec{r}, \vec{\Omega}, E, t)}_{\text{II}} + \underbrace{\Sigma_t(\vec{r}, E, t)\phi(\vec{r}, \vec{\Omega}, E, t)}_{\text{III}} = q(\vec{r}, \vec{\Omega}, E, t) \tag{2.6}$$

Term I represents the variation in time of the neutron flux at position $\vec{r}$, direction $\vec{\Omega}$ in the solid angle $d\vec{\Omega}$ and kinetic energy $E$ in the range $[E,\ E + dE]$ at the instant $t$. Term II is the net flux between the neutron flux that arrives at point $\vec{r}$ with direction $\vec{\Omega}$ in $d\vec{\Omega}$ and energy $E$ in the range $[E,\ E + dE]$ at the instant $t$, and the neutron flux that leaves position $\vec{r}$ with direction $\vec{\Omega}$ in $d\vec{\Omega}$ and energy $E$ in the range $[E,\ E + dE]$ at time $t$, when term II is positive the balance of the angular flux is negative. Lastly, term III represents the removal of neutrons from the angular flux due to any reaction of the neutron with a target. In Equation 2.2 the position vector $\vec{r}$ is three-dimensional and the direction vector $\vec{\Omega}$ is bi-dimensional, resulting in a seven-dimensional equation. In some representations, $\vec{\Omega}$ and $E$ are substituted by the three-dimensional velocity vector $\vec{v}$.

It is possible to perceive that the neutron transport equation is, stating in a simple manner, a neutron balance equation. Nevertheless, solving this nonlinear seven-dimensional equation is still an open challenge and subject of current research activity.

It is because of this difficulty that many numerical methods are used, where one of these methods is making use of a physical Monte Carlo simulation. This method is chosen because it can easily cope with complex boundary and domain geometries, but, because of its intrinsic stochastic nature, provides besides the mean also higher statistical moments. A particular property of Monte Carlo convergence is that the uncertainty of a sampled result decreases by the order of $N^{-1/2}$ (with $N$ the number of samples), and thus is independent of the number of dimensions of the problem, which in particular is an advantage for problems with higher dimensional variable spaces.

It is noteworthy that the physical Monte Carlo method is not used to solve directly the transport equation, differently to a mathematical Monte Carlo implementation, but instead it is used to simulate the microscopic phenomena that occur in a nuclear reactor and which have their correspondence in the transport equation. The nonlinear term of the double integral in Equation 2.3 arises from neutron emitting reactions, caused by the neutron flux. Therefore, in order to facilitate the numerical integration, physical Monte Carlo, i.e. simulation, is used to define reaction rates and neutron emission probabilities.

The results obtained from the stochastic method include the position, direction of travel and kinetic energy of each particle, the neutron flux, the neutron angular flux, among other physical quantities that may be obtained from the Monte Carlo simulation. As a consequence, the simulated spectral angular flux in a parametric form may be evaluated as a solution of the Boltzmann equation.

## 2.1 Simplifying Hypothesis of the Boltzmann Equation

Some simplifying hypothesis can be applied to Equation 2.2. This procedure will be done here for later use. The considered hypothesis reflect a scenario of stationary regime with isotropic neutron flux.

By considering a stationary regime and an isotropic flux, i.e. $\phi(\vec{r}, \vec{\Omega}, E, t) = \phi(\vec{r}, E, t)$, and by integrating both sides of Equation 2.2 by $\int_{4\pi} \bullet \, d\Omega$, the first two terms on the left hand side vanish and Equation 2.2 is reduced to 2.7.

$$4\pi \cdot \Sigma_t(\vec{r}, E, t)\phi(\vec{r}, \vec{\Omega}, E, t) = \int_{4\pi} q(\vec{r}, \vec{\Omega}, E, t) d\Omega \tag{2.7}$$

By considering in Equation 2.4 only two reactions that emit neutrons, fission and scattering, and by performing the integration in $d\Omega$ as shown in Equation 2.7, also by

removing the external neutron source, one obtains

$$\int_{4\pi} q(\vec{r},\vec{\Omega},E,t)d\Omega = 4\pi \int_0^\infty \phi(\vec{r},E',t) \sum_i \Big[ \nu\Sigma_{i,\,f}(\vec{r},E')\chi_f(E) + \Sigma_{i,\,s}(\vec{r},E')P_{i,\,s}(E' \to E)\Big]\, dE', \tag{2.8}$$

in which the probability $p_f(E' \to E, \vec{\Omega}' \cdot \vec{\Omega})$ of neutrons emitted by fission in an interval $dE$ in the vicinity of $E$ is given by 2.5, and $P_{i,\,s}(E' \to E)$ is the probability for a neutron with energy $E'$ to cause an emission in the interval $dE$ around $E$, after a scattering with nuclide $i$.

This scattered energy probability will later be computed numerically, by sampling the scattering procedure function for each nuclide. After these probabilities are found, the integral in the right hand side of Equation 2.8 will be evaluated. This will be done by a sub-routine for Monte Carlo integration. This differential scattering probability depends on the collision model and is further discussed in section 4.1.5.3.

Furthermore, an interpolating function can be found for the simulated spectral neutron flux. This function can then be used with Equation 2.7, to asses if it is a good candidate for the spectral neutron flux. Note that the left hand side of this equation represents the removal of neutrons from the neutron flux, whereas the right hand side represents the neutrons emitted into the flux. By the balance between both sides, one can assess how well the parametrized flux fits as a Boltzmann solution.

# 3 INTERACTIONS OF NEUTRONS WITH MATTER

Neutrons interact with matter in two ways. Scattering reactions and absorption reactions. Scattering reactions can occur with or without the formation of a compound nucleus as an intermediate state, while absorption reactions require the formation of the compound nucleus. Compound nucleus is defined by the intermediate state of the reaction, in which the nucleus is represented by the combined mass of the neutron and the nucleus. In reactions that involve a compound nucleus the neutron leaving the nucleus is not necessarily the same that has started the reaction [Feshbach, 1992].

A scattering always results in a neutron leaving the struck atom. An absorption, however, can result in different outcomes, such as the emission of gamma radiation ($n$, $\gamma$), the ejection of an alpha particle ($n$, ${}^{4}_{2}He^{+2}$), the ejection of a proton ($n$, $p$), or fission ($n$, $f$), among others [Glasstone and Sesonske, 1994]. Of these, ($n$, $\gamma$) and ($n$, $f$) are the dominant reactions for thermal reactors, while the remainder are small or even spurious reactions only.

In all reactions the following quantities must be conserved. *a)* Number of nucleons. *b)* Number of electric charges. *c)* Energy. *d)* Momentum. *e)* Angular momentum.

## 3.1 Neutron Scattering

Scattering reactions of a neutron with a nuclide are the ones responsible for the slowing down of neutrons in a thermal nuclear reactor, and thus of vital importance for this type of reactors. These interactions can be classified as elastic or inelastic scattering whether the kinetic energy is conserved in a reaction, or some part of the energy is converted into excitation energy of the target nucleus.

Scattering interactions can occur in different manners. The two main ways are resonant (or compound nucleus) scattering and potential scattering.

Resonant scattering occurs at higher energies in the vicinity of a resonance. In this kind of scattering the neutron and the target nucleus form a compound nucleus and later a neutron is expelled, leaving the nucleus in its ground state (elastic resonant scattering) or in an excited state (inelastic resonant scattering).

Potential scattering can occur in the whole energy range relevant for nuclear reactor neutrons and is attributed to peripheral scattering [Lamarsh, 1966; Reuss, 2008]. In this

form of scattering the neutron is scattered by its interaction with the strong nuclear force of the nucleus and both particles can be treated as in a classical particle collision due to the short range of the nuclear force ( $1\,fm$ or equivalently $10^{-15}\,m$). The larger part of the slowing down of neutrons in a thermal reactor comes from elastic scattering [Glasstone and Sesonske, 1994].

## 3.2 Elastic Scattering

Elastic scattering can be determined by considering it as a collision of classical particles, in which the energy and momentum of the system composed by the two colliding particles are conserved during the process. This implies that the participating nucleus is in its ground state after scattering, and there is no posterior emission of $\gamma$ radiation.

To find the energy of the neutron after an interaction, one needs to solve simultaneously the momentum and energy conservation equations, as these are the governing equations for this situation. Different scattering models were created and compared with respect to their computational efficiency and their validity in the stationary Boltzmann equation evaluation.

In order to clarify the problem we introduce three different reference frames, the laboratory *Lab* frame, the target frame, and the center of mass *CM* fame. The laboratory frame is the one used to measure the neutron energy, to compute its tracking and generate the problem geometry. The target frame is the system in which the target nucleus of a reaction is at rest. And finally, the center of mass frame is used to compute the energy and momentum conservation equations for the scattering reaction, it is also in this reference system that the scattering angle is defined.

The general problem can be reduced to the two-dimensional scenario represented by Figure 3.1 in the center of mass frame. In Figure 3.1, $v_{CM,1}$ and $v'_{CM,1}$ represent the speed of the neutron respectively before and after the scattering, and $v_{CM,2}$ and $v'_{CM,2}$ represent the speed of the target nucleus before and the scattering, respectively. The scattering angle in the *CM* frame is given by $\theta$, and its distribution is a function of target nucleus and energy.

In a general formulation, the kinetic energy of the neutron, measured in the center of mass frame, after the scattering is given by Equation 3.1, this equation can be achieved by applying energy and momentum conservation equations. In Equation 3.1 $E_{CM,1}$ and

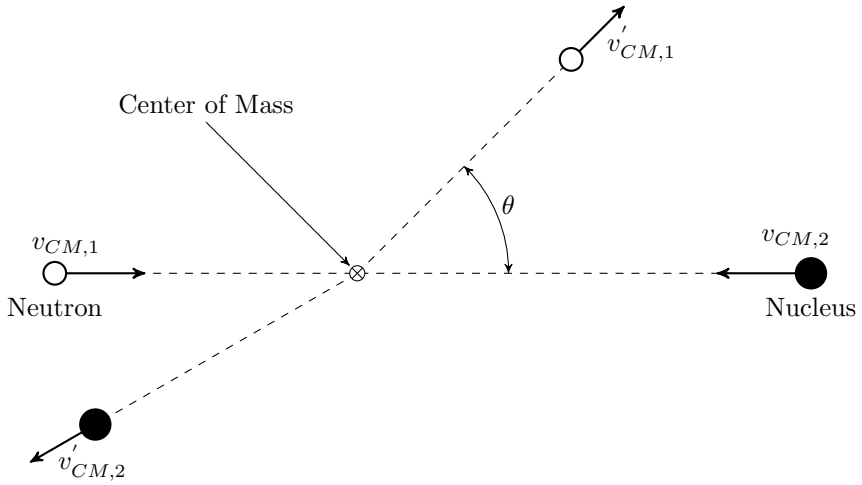


Figure 3.1 – Scattering in the center of mass system.

$E'_{CM,1}$ are the neutron energy before and after collision, $E_{CM,2}$ is the target nucleus energy before the collision, and $A$ is the target nucleus atomic mass. Note that the energy of the particles is computed in the center of mass frame.

$$E'_{CM,1} = \frac{(E_{CM,1} + E_{CM,2})A}{A+1} \tag{3.1}$$

In order to simplify the procedure, the hypothesis that the target is stationary in the laboratory frame can be used if the following assumptions are considered to be true. *a)* The nuclei are at rest relative to the neutron. *b)* The nuclei are not bound in a molecule or a solid [Glasstone and Sesonske, 1994].

These hypothesis are valid for high energy neutrons. According to Sunny et al., 2012 for epithermal neutron energies, i.e. from the order of a few $eV$ to the order of hundreds $eV$, the free gas model should be used in order to include the thermal motion of the scattering targets. In the thermal range, i.e. up to a few $eV$, the scattering matrices $S(\alpha, \beta)$ should be used to consider thermal motion and molecule binding effects. In this work both the fixed target hypothesis and the free gas model are implemented. However, the $S(\alpha, \beta)$ scattering matrices is not implemented and, therefore, the effects of molecular binding energy are not accounted for.

For light nuclei and low collision energies the scattering in the *CM* frame is a S-wave scattering, i.e. isotropic. However, as is shown by Lamarsh, 1966, as the target nuclei grow heavier or the collision energies become higher, the differential scattering cross section is no longer constant for all $cos(\theta)$, that means that the scattering is no longer

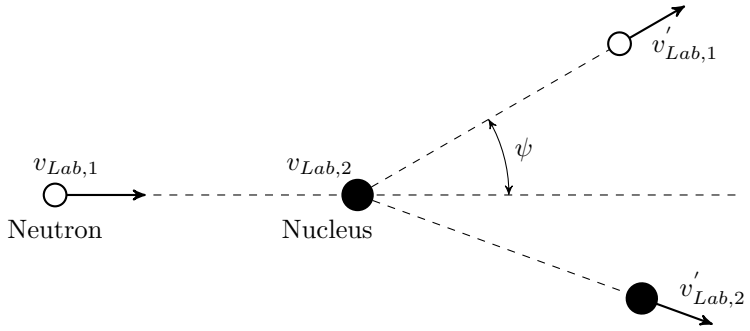


Figure 3.2 – Collision in the laboratory system.

isotropic and other partial wave functions should be taken into account.

When the kinetic energy of a neutron is of the same order of magnitude of the chemical binding energy of the atoms of the molecule involved in the collision (of the order of $eV$), the nuclide that plays a part in the scattering reaction is no longer considered free, but the molecule as a whole participates in the reaction. Consequently, new effects shall be taken into account such as new resonances due to collective modes, and inelastic effects.

### 3.2.1 Fixed Target Hypothesis

Should the neutron be in the moderating region and have a kinetic energy higher than the thermal region, i.e. of an order greater than $eV$, then the target nucleus can be considered stationary in relation to the neutron in the laboratory system. This condition already satisfies the prerequisite that the target nuclei are not bound in a molecule or a solid, for the incident neutron energy is large compared to the energy of chemical binding (also of the order of a few $eV$) and the nuclei can be considered as free. In the further the kinematic procedure for high energy neutrons is sketched.

A representation of this scenario is shown in Figure 3.2, in which the neutron has $v_{Lab,1}$ as the magnitude of its velocity vector before the collision and $v'_{Lab,1}$ after. The nucleus has a speed of $v_{Lab,2} = 0$ prior to the collision and $v'_{Lab,2}$ afterwards. The magnitude of the velocity vector of the center of mass of both particles in the *Lab* system is given by $v_{CM}$ and it is constant, due to conservation of momentum.

With the target stationary, the speed of the center of mass of the system $v_{CM}$ is given by

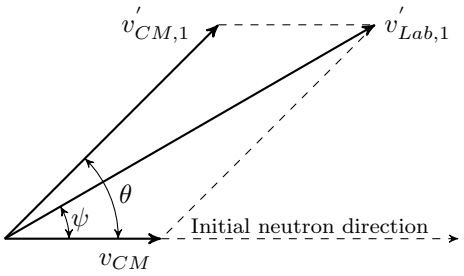


Figure 3.3 – Transformation between laboratory and center of mass system.

$$v_{CM} = \frac{v_{Lab,1}}{A+1} \tag{3.2}$$

in which $A$ is the mass number of target nucleus. Using Equation 3.2 and changing to the center of mass reference frame, results in a speed $v_{CM,1}$ for the neutron given by

$$v_{CM,1} = \frac{A\, v_{Lab,1}}{A+1} \tag{3.3}$$

and a nucleus has a speed of

$$v_{CM,2} = v_{CM} \tag{3.4}$$

prior to the collision.

From Equations 3.3 and 3.4, it is possible to arrive at the solution presented in Glasstone and Sesonske, 1994 and shown in Equation 3.5.

$$\frac{E'}{E} = \frac{v'^{2}_{Lab,1}}{v^2_{Lab,1}} = \frac{A^2 + 2\,A\,cos(\theta) + 1}{(A+1)^2} \tag{3.5}$$

in which $E$ is the energy of the neutron in the *Lab* system before the collision, $E'$ is the energy of the neutron in the *Lab* system after the collision and $\theta$ is the scattering angle measured in the *CM* system and that is in the plane that contains both vectors of incident direction and scattered direction of the neutron.

The scattering angle $\theta$ is measured in the *CM* frame. Its relation with the scattering angle $\psi$, measured in the *Lab* frame, is shown in Figure 3.3 and, for the target at rest hypothesis, is given by Equation 3.6.

$$cos(\psi) = \frac{A\,cos(\theta) + 1}{\sqrt{A^2 + 2\,A\,cos(\theta) + 1}} \tag{3.6}$$

in which $\psi$ is the angle of scattering measured in the *Lab* system and that is in the plane that contains both vectors of incident direction and scattered direction of the neutron. It must be highlighted that this procedure is done in the plane of scattering, it is still necessary to return to the three-dimensional space, to define the outgoing neutron direction. Since the target nucleus velocity is represented by a null vector, there are infinite possible scattering planes. The procedure to treat this is presented in Subsection 4.1.5.

A comment is in order here, it is possible to perceive that in Equation 3.5 the neutron can only lose energy and thus only down-scattering is defined. This is in agreement to our hypothesis that the neutron has far greater kinetic energy than the nucleus. Nevertheless it is known that the neutron can gain energy in a scattering reaction if it belongs to the thermal distribution. More specifically, this effect would only be appreciated with neutrons whose kinetic energy is of the same order of magnitude as the thermal energy of the nuclei, i.e. of the order of $eV$.

Using only the hypothesis of target at rest introduces a bias towards decreasingly smaller neutron energies. Note that by the model of Equation 3.5 a single scattering with a proton is sufficient to bring, in a head-on collision, even a fast neutron almost to a halt. In order to correct these inconsistencies, the present work distinguishes fission, intermediate and thermal distributions, and modifies the treatment given to the neutron, based on the distribution to which the neutron belongs. Thus it is considered that neutrons belonging to the fission and intermediate distributions have high enough energies so that the target at rest model can be used, and the neutrons of the thermal distribution have energies that maintain the thermal spectrum, so that no interactions have to be calculated explicitly, because the resulting spectrum is preserved and known.

### 3.2.2 Free Gas Model

In the free gas model, it is considered that the scattering target is not stationary, but instead it does have a thermal motion. This implies that the the simplification of Equation 3.5 is no longer valid, and Equation 3.1 must be used.

The scattering calculation for this model involves: *a)* Project the velocity vectors from the *Lab* frame to the *CM* frame. *b)* Use Equation 3.1 to define a new energy for the neutron in the *CM* frame. *c)* Find the outgoing neutron direction, by rotating it by $\theta$ in the *CM* frame. *d)* Project the velocity vector back to the *Lab* frame.

This model is not as straightforward as the fixed target model to define the outgoing neutron energy, because it cannot simplify the matricial operations. It does, however, present some advantages over the previous model, that includes the fact that it can define a single plane of scattering for each reaction, and it does not need to reinstate energy to the neutron in order to respect the thermal equilibrium distribution.

### 3.3 Inelastic Scattering

Inelastic scattering is a scattering reaction in which the nucleus is left at an excited state after the collision. Therefore, part of the kinetic energy of the particles involved in the interaction is transformed into excitation (internal) energy of the nucleus above its ground state. This energy is later emitted by the nucleus as $\gamma$ radiation in order to return to its ground state. The $\gamma$-ray must be considered in order to solve momentum and energy conservation equations.

Since inelastic scattering leaves the nucleus in an excited state, in order for it to occur, the kinetic energy of participating particles must be, at least, equal to the energy of this first excited state. This represents an energy threshold so that inelastic scattering can be possible. For heavier nuclides this threshold can be quite small, of the order of 44 $keV$ for U-238, and becomes much higher for lighter atoms, about 6.42 $MeV$ for oxygen, and it does not even occur in hydrogen [Lamarsh, 1966; Glasstone and Sesonske, 1994]. For this reason inelastic scattering is more prominent in heavier nuclei. Exceptions are the magic nuclei, that behave as light nuclei.

Inelastic scattering does, however, occur at low energies (below a few $eV$). But this does not happen through the formation of a compound nucleus as the aforementioned process, it also does not leave the nucleus in an excited state. This type of inelastic scattering happens when the kinetic energy of the scattered neutron is so low that the target atom is no longer considered as a free atom, but bound in a molecule or in a solid. What takes place is then the interaction of the neutron with the quantum states of vibrational and rotational motion of the molecule (or just vibrational for a solid state). The neutron can then gain or lose energy with such interaction, as a result of a change in these motion states.

The change in the energy of an inelastically scattered neutron by a point-like target depends on the model. For the free gas model, the excitation energy must be accounted

for in the momentum and energy conservation equations, with the rest of the procedure remaining the same. For the fixed target model, it is given by Equation 3.7, and the relation between the angles $\theta$ and $\psi$ is given by Equation 3.8.

$$\frac{E^{'}}{E} = \frac{v^{'2}_{CM,1}}{v^{2}_{CM,1}} = \frac{\gamma^2 + 2\,\gamma\,cos(\theta) + 1}{(A+1)^2} \tag{3.7}$$

$$cos(\psi) = \frac{\gamma\,cos(\theta) + 1}{\sqrt{\gamma^2 + 2\,\gamma\,cos(\theta) + 1}} \tag{3.8}$$

in which the $\gamma$ parameter is given by Equation 3.9.

$$\gamma = A\sqrt{1 - \frac{A+1}{A}\frac{Q}{E}} \tag{3.9}$$

In Equation 3.9 $Q$ is the excitation energy of target nucleus [Reuss, 2008]. Should $Q$ be equal to 0, $\gamma$ is reduced to $A$.

If neutrons that suffer inelastic scattering have energies that are able to excite only a few energy levels of the nucleus, the outgoing neutrons will have a spectrum that shows distinct energy groups, one for each excitation level. However, if the neutrons have energies that excite the higher and closely spaced energy levels the departing neutrons will present a continuous spectrum, in addition to the discrete one [Lamarsh, 1966].

As presented in Lamarsh, 1966 this spectrum can be defined by Equation 3.10.

$$P(E \rightarrow E^{'}) = \frac{E^{'}}{T^2} e^{\frac{E^{'}}{T}} \tag{3.10}$$

in which $T$ is the nuclear temperature and is given approximately by Equation 3.11.

$$T = 3.2\sqrt{\frac{E}{A}} \tag{3.11}$$

in which $T$, $E$ and $E^{'}$ are in $MeV$, $A$ is the mass of target nucleus and the constant 3.2 is in $MeV^{1/2}\,a.m.u.$. It must be remarked that Equation 3.11 is not a very good approximation, specially for magic and near-magic nuclei. In future Monte Carlo implementation experimental data for inelastic scattering will be parametrized and included in the nuclear reactions library. It is noteworthy that the inelastic scattering cross-section represents only a small fraction of the total scattering cross-section, and thus may be considered a small correction.

### 3.4 Fission

In nuclear fission, a heavy nucleus splits into two lighter nuclei and releases energy, since the sum of binding energies of both lighter nuclei is smaller than the binding energy of the original nucleus. Although spontaneous fission can occur in a nuclear reactor, it is a rare event and, therefore, only fission resultant from a neutron absorption will be considered.

For fission to happen an energy threshold for the compound nucleus must be exceeded. This critical energy can be surpassed just by the binding energy of the last neutron, in which case the nucleus is called fissile and it can be fissioned by neutrons of zero kinetic energy. Should the compound nucleus critical energy be greater than the binding energy added by the absorption of the neutron, the remaining energy must be provided by the kinetic energy of the neutron, in which case the nucleus is called fissionable.

This difference of fissile and fissionable nuclei is shown in the cross sections for these interactions. For fissionable nuclei the cross section is zero up to the difference between the energy threshold and the binding energy of the last nucleon. In the case of fissile nuclei the fission cross section exists for all energies, in fact, it grows towards lower energies according to $1/\sqrt{E}$ and present resonances at higher energies.

Starting with a heavy nucleus, fission results in two lighter nuclei. The atomic number is divided between these fission products, but hardly ever is the number of protons equally divided, i.e. fission is asymmetric. These resulting nuclei are usually in an excited state and will decay in an order of time greater than the time-scale of fission. The distribution of fission products is dependent on both the nuclide that is being fissioned and the neutron energy.

Fission is accompanied by the release of neutrons. The number of neutrons emitted in each fission varies, and the mean depends on both the nuclide and the colliding neutron energy. These neutrons are emitted in two different time-scales, and thus are called prompt and delayed neutrons. Prompt neutrons are the ones emitted in a time-scale of $10^{-14}s$ or less [Glasstone and Sesonske, 1994]. These prompt neutrons represent more than 99% of the neutrons emitted by fission reactions.

The prompt fission spectrum for fission of U-235 is given by Equation 3.12, this is a probability distribution function for the energies of prompt neutrons. Similar data can also be found for other fissionable nuclei, e.g. plutonium.

$$\chi(E) = 0.453\, e^{-1.036\ MeV^{-1}\, E} \sinh \sqrt{2.29\ MeV^{-1}\, E} \tag{3.12}$$

here the energy $E$ is given in units of $MeV$.

As stated before, fission products can stay in an excited state for a long time before decaying. In their decay chains towards the stability line, these products can decay via neutron emission. The order of time after a fission occurrence these neutrons are emitted is of the order of seconds, a time-scale much greater than the time-scale of prompt neutrons. Delayed neutrons represent less than 1% of neutrons originated by fission, e.g. 0.65% for thermal fission of U-235.

It is common to group delayed neutrons according to their precursors nuclides, i.e. the fission products that will eventually decay and give birth to a neutron. These precursors have a half-life that can vary from about 55 $s$ to the order of $10^{-1}$ $s$. Contrary to prompt neutrons, that have a continuous distribution that spreads along several $MeV$ for their energy, delayed neutrons have much better defined energies that depend on their precursor and are of the order of $10^{-1}$ $MeV$ to 1 $MeV$.

## 3.5 Radiative Capture

Radiative capture reactions are one possible outcome of the absorption reaction that captures one neutron in a nucleus, and effectively removes it from the neutron flux. After the formation of a compound nucleus, the increase in the nucleus internal energy due to the neutron binding energy and kinetic energy is evenly divided amongst all nucleons in such a way that the nucleus can stay at this excited state for a long amount of time (in comparison to the time it takes to form a compound nucleus). After some time this excited nucleus decays via emission of gamma-rays.

## 3.6 Other Interactions

Other interactions of neutrons with nuclides are possible, such as the $(n, 2n)$ and $(n, 3n)$ reactions. These, however, have small cross sections when compared to aforementioned reactions and can be disregarded in a first approach when treating thermal reactors. The author of this thesis is aware that, for a full simulation of the reactor core including actinides and fission products these reactions shall be included, specially if ki-

netics aspects are of interest, and will be accounted for in a future work, but are neglected in the present stage of the simulator development.

# 4 SIMULATOR DEVELOPMENTS

The Monte Carlo simulator is composed of three distinct C++ programs. The first of these is a program that receives a cross-section file, taken from the ENDF library, and outputs sectionally continuous functions in C++ source code, built as to minimize comparative operations (a better description of this procedure can be seen in A.1). The second program is the one that computes the simulation and/or its data assessment, its main functionality is explained in section 4.1. The final program aims to evaluate the parametrization of the spectral neutron flux as a candidate solution to the Boltzmann neutron transport equation, its computations and simplifications are described in section 2.1.

The simulator is continuous in all seven dimensions of the Boltzmann equation, creating bins only during data assessment in order to reduce the size of the evaluated data, and thus adequate the set of data to the plot resolution. Also, the underlying interaction and tracking philosophy is similar to the GEANT paradigm and hence turns feasible an insertion as a module in this simulation platform.

## 4.1 Monte Carlo Simulation

The C++ Monte Carlo simulator in development is written according to parallel computation standards. Since the main information resulting from the simulation, and used for data assessment, is the ensemble of neutron reactions, this task is straightforward. The simulation is composed by the sum of various independent executions. Each of which is executed in a separate thread, with a module managing when threads are available and enchaining a new execution to it.

These executions start with a parcel of the total neutron histories to be simulated, and it is limited to the total predefined number of Monte Carlo steps. For tallying reasons linked to computer hardware constraints, these were segmented in intervals of a given number of steps each, i.e. after a given number of steps the simulation reaches a checkpoint, where it is halted and the respective data set is saved. The subsequent interval then used this data as the initial condition for its following steps. This way, by increasing the frequency of saving the data set maintains memory usage within an operational mode, but in turn increases the time required for disk writing and reading operations. The size

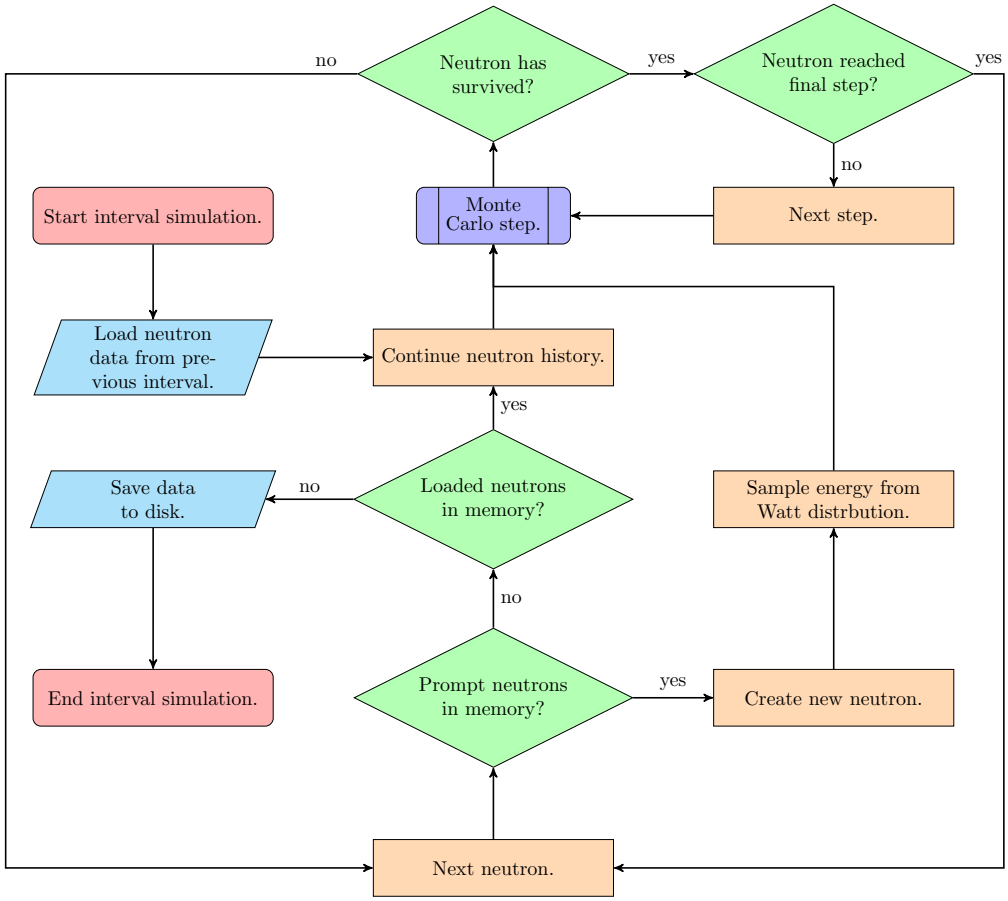


Figure 4.1 – Simulation flowchart with its principal instances.

of these segments can be tailored to optimize the execution time for a given hardware configuration.

To better perform in the saving and loading of data, a particular data structure was created. This structure was then used to write and read data to and from binary files, an operation whose speed excels that of outputting and inputting formatted data. It also results in smaller archive for the same amount of information. More details about this procedure is given in appendix B.

A flowchart showing the main scheme of the aforementioned intervals is shown in Figure 4.1. The beginning and end of each interval is marked by loading the neutrons from previous steps and saving their information for the succeeding ones. If the interval under consideration is the first one, then, by default, a purely fission population that is homogeneously distributed inside the reactor domain is generated. However, any initial scenario is possible, including using results from previous simulations.

Note that the chart in Figure 4.1 states that fission produced neutrons take prece-

dence over the neutrons from previous intervals that are already loaded in the memory, the reason for this is to prevent the growing of the buffer vector that stores fission information, and thus prevent a higher memory usage. Neutrons are iterated from the initial interval Monte Carlo step to the last, in which case this information is stored in RAM until there are no more neutron histories to compute, thereupon this data is saved to disk. Neutron born from fission continue from the next Monte Carlo step from which they were created and are also stored for subsequent intervals if the fission happens at the last step.

The neutron class contains all information needed for the simulation of tracking and interaction with matter. Not only that but it also contains the tags that characterize the particle and allow for the proper tally of histograms in the data process part of the data evaluation program. A scheme of the neutron class with its principal members is shown in Figure 4.2.

Tags can be created according to the desired results. They do not influence the running of the simulation, but are used for data processing. The user can introduce new tags, as well as the methods for accounting them.

### 4.1.1 Distributions

One novel method used in this program is the division of the neutron population according to three different probability distributions. Two of them have known shape, but unknown population, these are the Maxwell-Boltzmann spectrum for thermal neutrons, and the Watt spectrum for fission born neutrons. The third is an intermediate distribution of *a priori* unknown shape and unknown normalization. It is noteworthy that these distributions can coexist for a given energy, in such a way that stochastic criteria are used to defined to which distribution a neutron belongs to.

### 4.1.2 Interaction Model

The program allows the user to choose the scattering model. This can the fixed target model or the free gas model, as discussed in Section 3.2. In order to maintain consistency, this also changes the way cross sections are calculated.

Due to the way cross sections are measured [Brown et al., 2018], they must be measured in the target frame of reference. For the fixed target model, this implies that both laboratory and target frames are the same. For the free gas model, the target has

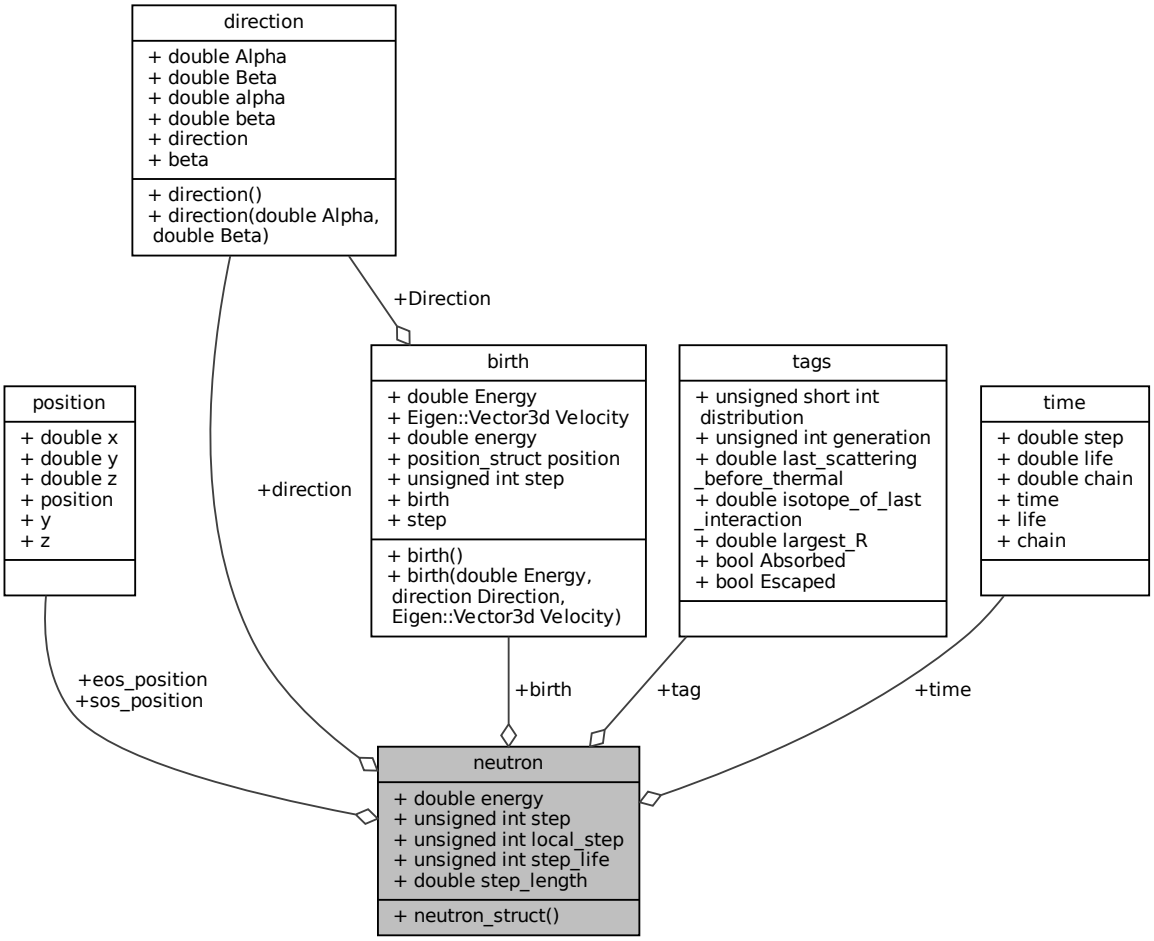


Figure 4.2 – Members of the neutron class.

a thermal motion and a transformation between the reference systems is required. Apart from an additional matrix algebra, this results in the fact that, the free gas model does not require a Doppler broadening correction, since it already accounts for the target thermal motion. As opposed to the fixed target model, in which a correction is required to properly evaluate the cross section at a temperature different from the one the measurement was performed.

### 4.1.3 Program Initialization

Before the simulation can properly begin, the program needs to map the geometry to the elements that compose each region, as well as their properties. It is by this procedure that the total macroscopic cross section along the neutron's path of travel can be found. It also allows for the definition of the target nuclide and the neutron interaction.

The program first constructs objects of the nuclide class. This class maps the microscopic cross section functions of each isotope to its atomic mass and identifier.

Then by using the available nuclides, an object of the element class is constructed. This class creates an element by selecting the element's isotopes and attributing them an isotopic abundance. If an element has more than one isotopic distribution, new objects must be created to acknowledge this, e.g., enriched and depleted Uranium have different proportions of $^{235}U$ and $^{238}U$, so that one shall define one Uranium element per enrichment value.

Now that elements are defined, the program initializes objects of the molecule class. Here the molecule class maps the stoichiometric proportion of each element that composes a given molecule.

Finally the objects of the region class are created. By definition, each region objects correspond to an homogeneous mixture of molecules in a given volumetric proportion. The region objects are then assigned to the different volumes of the problems geometry.

These classes have methods that allow for two main functionalities. The first is, given the energy of a neutron in a certain region, compute the region's total macroscopic cross section. The second is, defining that the Monte Carlo step ends within the region, to select the participating molecule, element, nuclide, and the neutronic reaction in a stochastic manner.

### 4.1.4 Monte Carlo Step

All the seven parameters that are present in the Boltzmann transport equation are considered, where the path that the neutron will travel is selected stochastically, and the position of a reaction is determined. A comment is in order here, the tracking and the interaction scheme was optimized in the sense that each Monte Carlo step has an interaction, which increases computational efficiency, but at the cost of loosing a unique relation between Monte Carlo step and corresponding time interval. Thus, a Monte Carlo step may be related only to an average of a time interval distribution, that may be reconstructed from the tallies. After the displacement of the neutron its position is checked in order to evaluate whether it still remains in the reactor core volume or whether it escaped, where in the later case the history of the neutron ends and a new neutron is selected from the stack. Finally, the type of neutron interaction is selected. This is based on both region and neutron energy, and the process is stochastic.

In the case of a radiative capture the procedure is the same as for escape, the neutron's history simply ends, and a new neutron is chosen from the stack. In case that fission occurs, a number of newly generated neutrons is chosen randomly and their positions are that of the fission reaction, which then are added to the stack. The history of the fission inducing neutron is then ended. In the case of scattering (the details will be shown later in this section) the energy and the direction angles are updated for the next step. Should the step reach a checkpoint, as mentioned above the seven transport variables are recorded and are used by the subsequent Monte Carlo cycle. The scheme of a Monte Carlo step is presented in Figure 4.3.

In the Monte Carlo step flowchart there are several instances in which data is saved, these represent the storage of information in buffers and not file access operations. These buffers are only emptied at the save operation depicted in Figure 4.1. Therefore, the loading and saving of files is done at the beginning and end of each interval, and file operations inside a Monte Carlo step are kept to a minimum.

The position in which a reaction will occur at the end of a Monte Carlo Step depends on the kinetic energy of the neutron, its position at the beginning of the step, the direction of movement and the total macroscopic cross sections of the chemical composition of the reactor core material along the trajectory. The final position of the track will then be determined by a stochastic selection for the length of the traveled path. To this

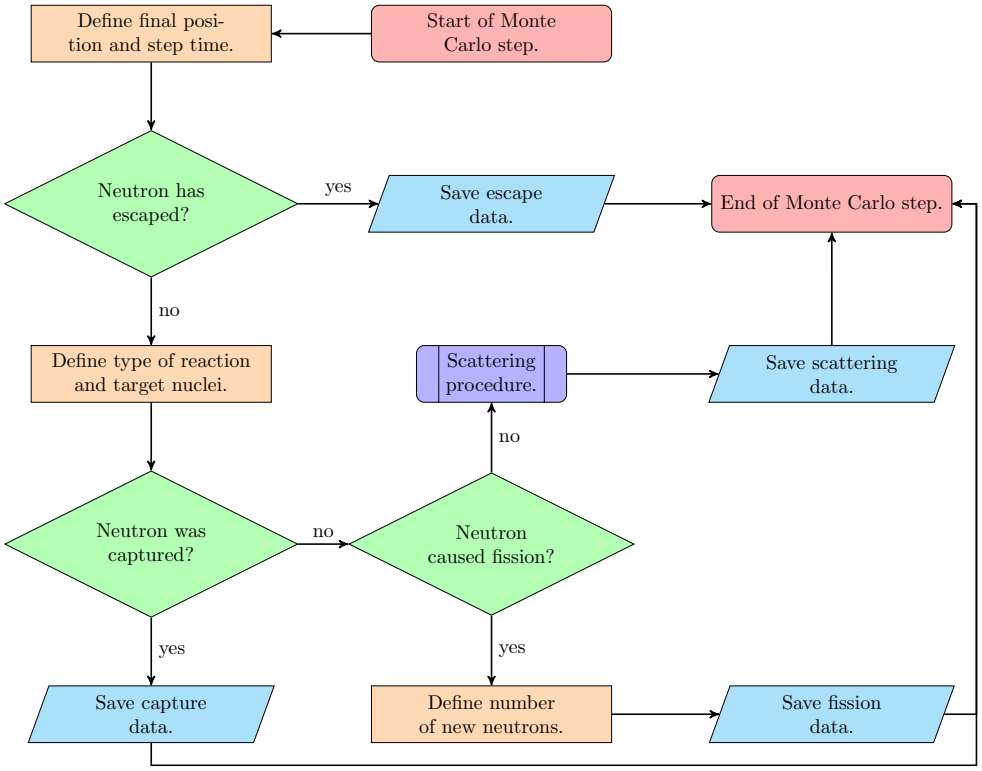


Figure 4.3 – Monte Carlo step flowchart with its principal instances.

end a multiple $S$ of the mean free path is generated by a random number, following a standard procedure $S = -\ln(1-a)$ and $a \in [0,1]$. Consequently the length of the path is $L = S\Sigma_t^{-1}$ where $\Sigma_t$ is the microscopic total cross section characteristic for the path. In case a neutron crosses the reactors boundary the exit point is computed and the tracking of this particle is terminated.

After the position of the interaction is defined, the target involved in the reaction is chosen. The type of interaction is selected by the generation of three random numbers, the first is to choose the molecule with which the neutron interacts. The molecule is chosen based on the proportion of its total macroscopic cross-section to the sum of the total macroscopic cross-section of all molecules. It is important to highlight that the uranium dioxide molecules ($UO_2$) are treated separately, according to the uranium isotope present ($^{235}U$ and $^{238}U$).

Subsequently the nuclide in the molecule is chosen, this time by the proportion that each nuclide's total macroscopic cross-section contributes to the molecule's total macroscopic cross-section. And lastly the interaction is chosen by the ratio of each interaction cross-section to the total one.

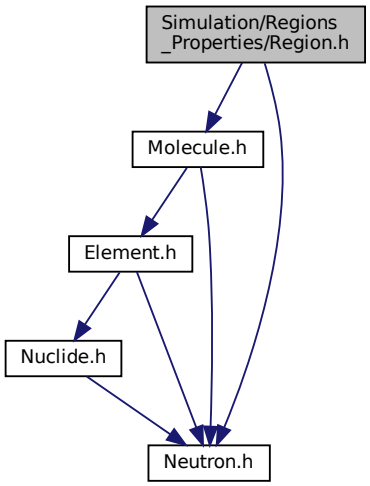


Figure 4.4 – Include graph of the region class.

This procedure can be visualized by the graph of include files, in which the dependency between the aforementioned classes is shown in Figure 4.4. There the dependency between region, molecule, element and nuclide classes becomes clear, where the previous always contains objects of the next. These objects are stored in maps and the classes implement a method for the stochastic selection of one of these elements, based on the macroscopic cross section of the respective molecule, element and nuclide, cascading down from the the homogeneous mixture of a region, to a selected interaction with a given nuclide.

If the chosen reaction is fission two stochastic operations are in order. The first one is to decide the number of neutrons born from fission, and the second is to define their energies. In order to define the number of neutrons from fission a random number $\nu$ is generated and will be equal to 2 or 3, the mean of this operation coinciding with the expected value of $\bar{\nu} = 2.48$ for fission induced by thermal neutrons in U-235. It is noteworthy that the huge bulk of fission reactions releases either two or three neutrons, so that, to a good approximation, only these two cases are taken into account. These neutrons have energies roughly in the range between $10^0\ MeV$ up to $10^1\ MeV$ as given by Equation 4.1. At the present state of developments no contributions due to delayed neutrons are considered, this pertinent issue will be included in the next version of the simulator.

$$\chi(E) = 0.453\, e^{-1.036\ MeV^{-1}\, E} \sinh \sqrt{2.29\ MeV^{-1}\, E}\ . \tag{4.1}$$

In case of a fission the position of the reaction, the number of released neutrons as well as information regarding the synchronization of the program are stored in a buffer. The histories of neutrons produced by fission are computed before the ones of already existing neutrons, this is done in order to free storage positions of the fission buffer, thus reducing the usage of random access memory.

### 4.1.5 Scattering

In case of scattering, a new energy and a new direction in agreement with energy and momentum conservation must be given to the neutron. A simplification of the program is that it considers the scattering as elastic and isotropic in the center of mass system. Strictly speaking, scattering is isotropic for low kinetic energies and small nuclei, however, as the collision energies become higher and/or target nuclei become larger, anisotropy increases.

A necessary feature of scattering not implemented yet is due to to the fact that approximately below 1 $eV$ instead of a single free nuclide one has to consider whether the atom is a constituent of a molecule or solid state, so that in the previous case molecular degrees of freedom such as rotation and vibration shall be considered, whereas in a solid state phonon degrees of freedom shall be taken into account. One used method to account for the effects of bound atoms is to use the scattering matrix [Williams, 1966], a procedure that is available in softwares like the MCNP6 and GEANT4 platform [Monk et al., 2017; Tran et al., 2018].

As already discussed in the previous chapter, two models for scattering reactions were implemented. They are the fixed target model, and the free gas model. The user must select which of these models to use. In the next subsections the technical particularities of the implementation of each model are discussed.

#### 4.1.5.1 Fixed Target Model

This model treats down-scattering only, i.e. energy loss of neutrons in their interactions with their respective targets. However, the closer neutrons approach thermal

energies also up-scattering is important, due to the thermal motion of the target nuclei which is no longer negligible in comparison to the neutron's kinetic energy. As soon as neutrons may be classified as thermal they are in equilibrium with the environment which allows to simplify the tracking and interaction procedure. Equilibrium implies conservation of the respective energy distribution, which from the microscopic point of view means that neutrons in the average gain as much kinetic energy in collisions as much as they loose, so that the only relevant stochastic quantity that shall be determined is the reaction rate. According to Bell and Glasstone, 1970, for large well-moderated reactors with homogeneous temperature, the Maxwell-Boltzmann distribution can be a good approximation for the thermal neutron spectrum. This, however, is not the case for heterogeneous reactors with large temperature gradients, in which a more detailed treatment is required.

The procedure to determine whether a neutron belongs thermal distribution is as follows. A random number between 0 and 1 is generated and by comparison to the Maxwell-Boltzmann cumulative distribution for the equilibrium temperature. Should the random number be grater than the cumulative distribution, then the neutron is considered to be in thermal equilibrium with the moderator. Neutrons that are part of the thermal population are assigned a new energy at each scattering, sampled from the Maxwell-Boltzmann probability distribution. This procedure is depicted in Figure 4.5.

As stated in the previous chapter, using the target at rest model, implies in the target having a null velocity vector in the *Lab* frame. This results in the existence of infinite possible scattering planes. In order to solve this, the procedure to find the new neutron direction after scattering is determined in a complete three dimensional fashion, although the cylinder symmetry would allow a reduction into a plane. Let the unit vector $\vec{\Omega}_i$ be the direction of the incoming neutron with $\alpha \in [0, 2\pi]$ and $\beta \in [-\pi/2, \pi/2]$ angles with respect to the laboratory reference frame, see Equation 4.2. For convenience one may construct one possible final direction $\vec{\Omega}_f^*$ as the one defined in Equation 4.2 shown in Figure 4.6.

$$\vec{\Omega}_i = \begin{pmatrix} \cos(\alpha)\ \cos(\beta) \\ \text{sen}\,(\alpha)\ \cos(\beta) \\ \text{sen}\,(\beta) \end{pmatrix}, \qquad \vec{\Omega}_f^* = \begin{pmatrix} \cos(\alpha)\ \cos(\beta+\psi) \\ \text{sen}\,(\alpha)\ \cos(\beta+\psi) \\ \text{sen}\,(\beta+\psi) \end{pmatrix} \tag{4.2}$$

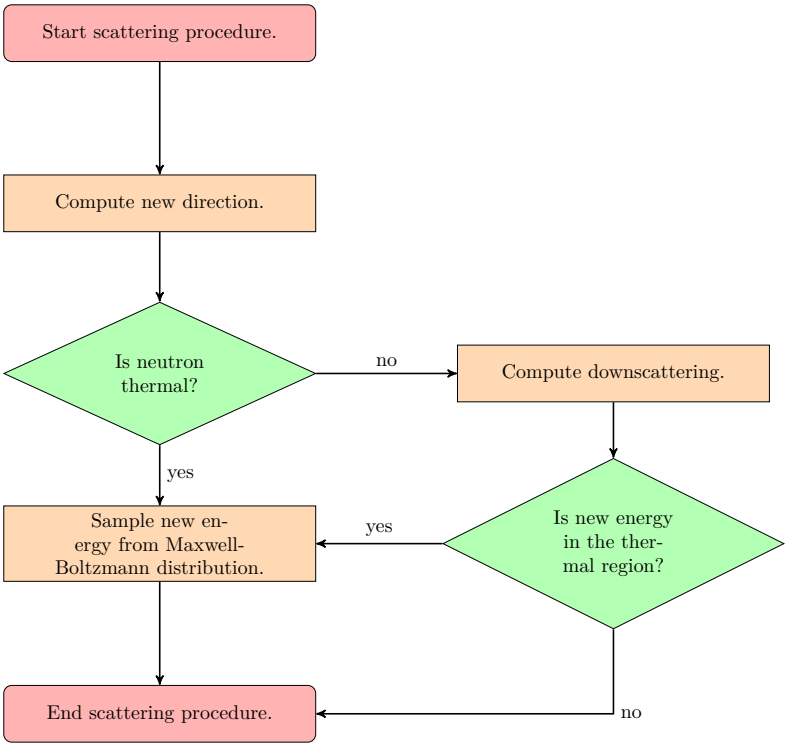


Figure 4.5 – Fixed target scattering procedure flowchart.

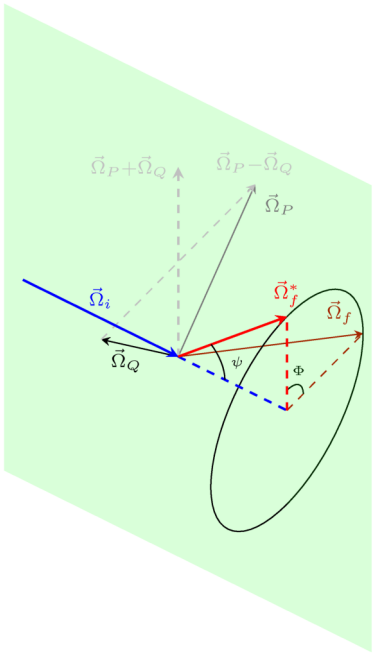


Figure 4.6 – Sketch of the neutron scattering scheme.

All remaining possible final vectors in agreement with cylinder symmetry may be generalized with two auxiliary orthogonal vectors $\vec{\Omega}_P$ and $\vec{\Omega}_Q$, that by construction are symmetrical on either side of the scattering plane defined by $\vec{\Omega}_i$ and $\vec{\Omega}_f^*$. The vector $\vec{\Omega}_P + \vec{\Omega}_Q$ lies then in the scattering plane, whereas $\vec{\Omega}_P - \vec{\Omega}_Q$ is perpendicular to the latter. The plane by $\vec{\Omega}_P$ and $\vec{\Omega}_Q$ is obtained by the solution of the system in Equation 4.3 and defines the rotation plane that contains the circle with all possible outcomes for the final direction $\vec{\Omega}_f$ of the neutron after scattering.

$$\begin{cases} \frac{\text{sen}\,(\psi)}{\sqrt{2}}\,(\vec{\Omega}_P - \vec{\Omega}_Q) = \vec{\Omega}_i \times \vec{\Omega}_f^* \\ \frac{\text{sen}\,(\psi)}{\sqrt{2}}\,(\vec{\Omega}_P + \vec{\Omega}_Q) = \vec{\Omega}_f^* - \cos(\psi)\,\vec{\Omega}_i \end{cases} \tag{4.3}$$

Therefore, $\vec{\Omega}_f$ is given by

$$\vec{\Omega}_f = \cos(\psi)\,\vec{\Omega}_i + \text{sen}\,(\psi)\,(\cos(\Phi)\,\vec{\Omega}_P + \text{sen}\,(\Phi)\,\vec{\Omega}_Q) \tag{4.4}$$

in which $\Phi \in [0, 2\pi]$ is a random angle.

It is noteworthy, that in the literature the problem is treated typically in two dimensions only, because of the scattering inherent cylinder symmetry. The present method makes use of a random angle in the center-of-mass system, but with three dimensional vectors. In this procedure two auxiliary vectors are created, one in the scattering plane $(\vec{\Omega}_P + \vec{\Omega}_Q)$ and one perpendicular to the latter $(\vec{\Omega}_P - \vec{\Omega}_Q)$ that define a plane orthogonal to the incident direction. The usage of the sum and the difference for the auxiliary vectors is due to the fact that two non-collinear vectors with equal length generate orthogonal vectors by their sum and difference. These two vectors span the plane that allows to represent the cylinder symmetry and utilize a random angle that may rotate the scattering plane into any other possible orientation compatible with the scattering kinematics.

#### 4.1.5.2 Free Gas Model

In the free gas model, the target nucleus has a thermal motion given by the temperature of the medium. It is given a random energy, sampled from the Maxwell-Boltzmann distribution, and a random direction taken from an isotropic distribution. Therefore, this model involves the solving the energy and momentum conservation equations for the collision of two particles moving in three-dimensional space.

A vector calculus approach was chosen to solve this problem. This was greatly

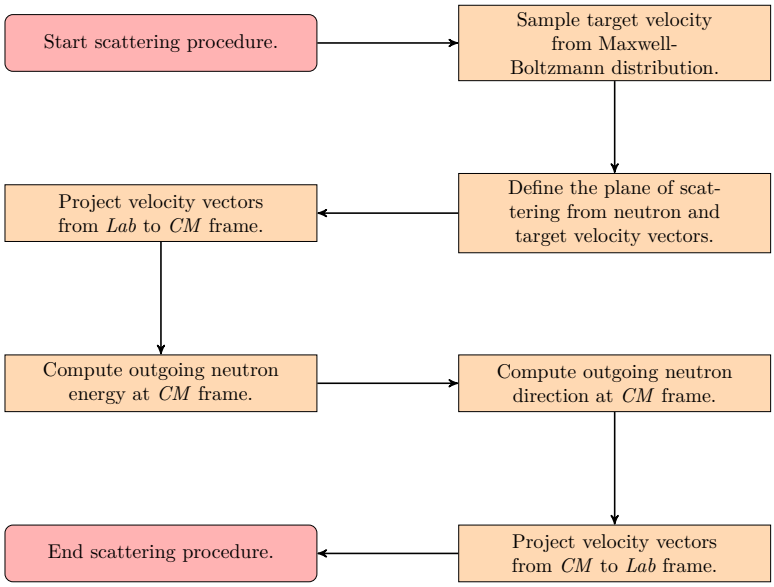


Figure 4.7 – Free gas scattering procedure flowchart.

facilitated and optimized by the use of the *Eigen* library [Guennebaud, 2011, 2013]. This is a C++ template library for linear algebra, that provides good performance with a simple syntax.

The scattering procedure begins by calculating the plane of scattering, defined by the cross product of the velocity vectors, and reducing the dimensionality of the problem to a two-dimensional space. Then we proceed to find the velocity of the center of mass of the system, so that we can transform the problem from the *Lab* frame to the *CM* frame. At the *CM* we calculate the outgoing neutron energy by using Equation 3.1. The outgoing direction, in the *CM* frame, is found by rotating the entire system by the scattering angle $\theta$. Finally we must return to the original three-dimensional *Lab* frame. This procedure is depicted in Figure 4.7.

This procedure does not require further correction of the neutron energy, since the thermal motion was applied to the target particle. It is possible to perceive that this allows not only for a decrease of energy of the neutron in the *Lab* frame, but also for its increase.

#### 4.1.5.3 Difference between Models

In order to better evaluate the particularities of each scattering model, and thus, provide a good basis for comparison between them, the simulation of the standalone

scattering procedure was performed $10^6$ times and incremented $100 \times 100$ logarithmic energy bins. This resulted in the differential scattering probability, i.e. surface plots that map an incoming neutron energy to a distribution of possible outgoing neutron energies.

The differential scattering probability for the target at rest model is shown in Figures 4.8 and 4.9. And for the free gas model in Figures 4.10 and 4.11. For each model the scattering of a neutron with both $^{238}U$ (Figures 4.8 and 4.10) and $^{1}H$ (Figures 4.9 and 4.11) was calculated.

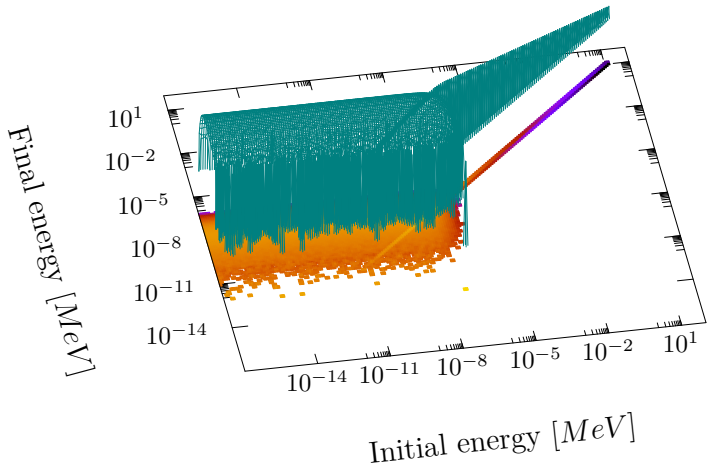


Figure 4.8 – Differential scattering probability with $^{238}U$ in the target at rest model.

In Figures 4.8 and 4.9 it becomes clear the implementation of up-scattering effects by the sampling of the Maxwell-Boltzmann distribution for thermal neutrons. This is specially apparent in the constant distribution for the final energy the neutrons have in a scattering reaction. In this model, even with a stochastic criteria to define the neutron as part of the thermal population, there is a clear indicative of the region (around the initial energy of $10^{-7}\,MeV$) where the thermal effects become dominant.

In contrast, Figures 4.10 and 4.11 do not present a fixed distribution probability for low energy neutrons, in fact, neutrons with low initial energies can still down-scatter, something not possible in the target at rest model. It is also noteworthy that there is no clear separation of the down-scattering dominated region from the region where thermal effects are the predominant mechanism (this effect can be more easily identified in Figure

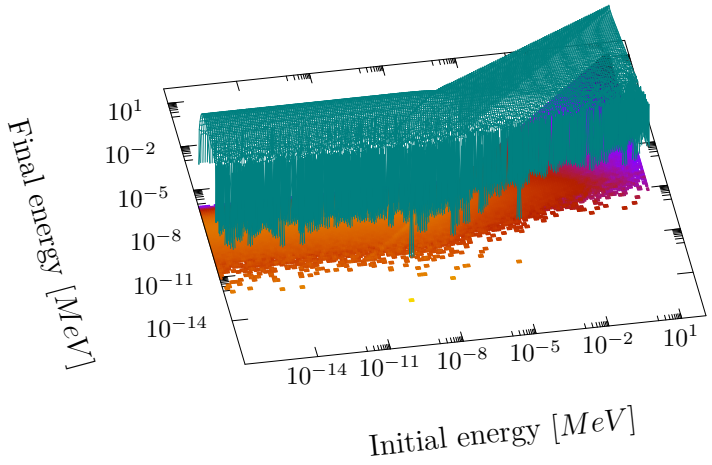


Figure 4.9 – Differential scattering probability with $^{1}H$ in the target at rest model.

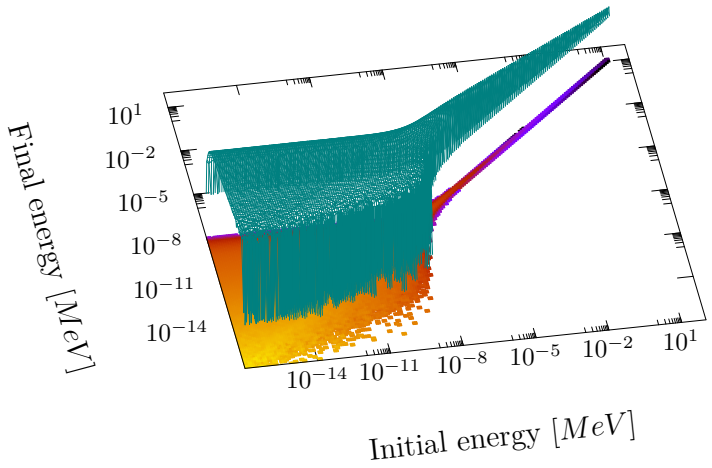


Figure 4.10 – Differential scattering probability with $^{238}U$ in the free gas model.

4.10).

In spite of their differences, in both models, the difference between a scattering with a light or heavy target is maintained. In both models the light Hydrogen nucleus can, in a few scattering reactions, bring the neutron to thermal energies, while for a heavier nucleus like Uranium, more scattering reactions are needed to slow down the neutron.

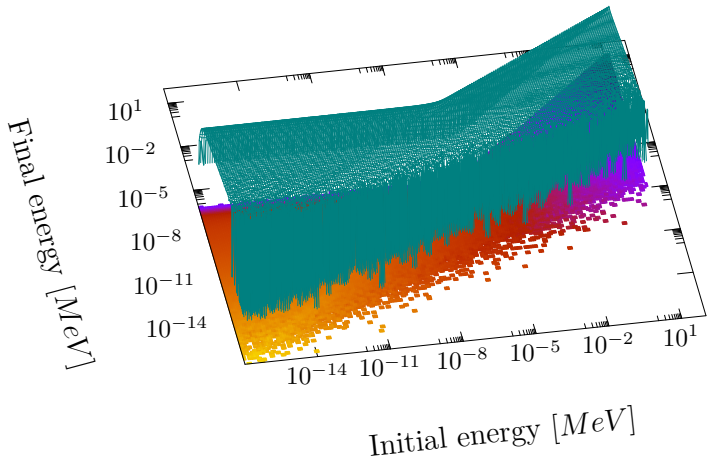


Figure 4.11 – Differential scattering probability with $^{1}H$ in the free gas model.

Another important topic is the computational statistics of each model. In Table 4.1 the mean computational time of a single scattering reaction is shown for each model.

Table 4.1 – Comparison of scattering models computational time.

| Target at Rest | Free gas |
|---|---|
| $6.60\,\mu s$ | $33.75\,\mu s$ |

Although conceptually simpler than the fixed target model (and allowing for simpler code), the free gas model, with its unique treatment of the whole energy spectrum, needs for all scattering reactions to be solved using matrix algebra, independent of the neutron energy. In fact, as is shown in Table 4.1, the free gas model computational time exceeds the fixed target model computational time by a factor greater than 4. Note also, that the free gas model also has a larger overhead in cross section computations.

## 4.2 Data Assessment

After the simulation is completed, the data assessment part is responsible for joining the information and computing the desired results. The results can be found by accumulating data of interest along the histories of the simulated neutrons, and by filtering the information of interest according to tags that are modified during the simulation.

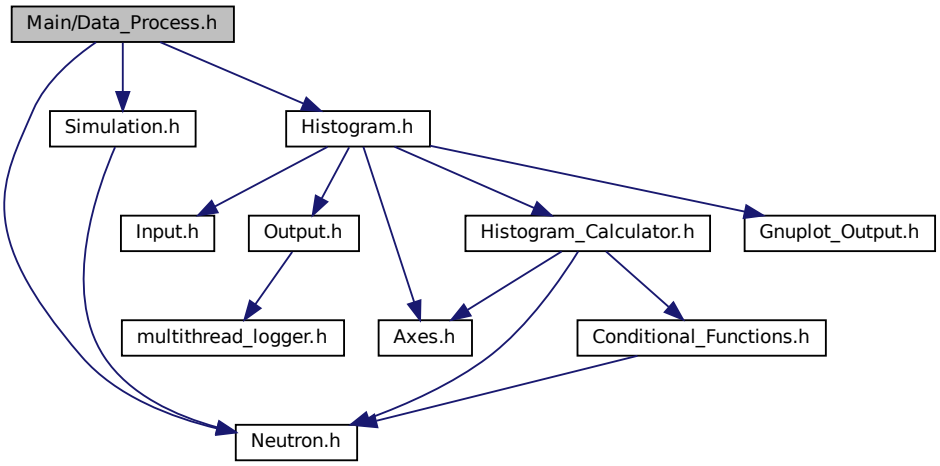


Figure 4.12 – Include graph of the data process functions.

Custom tags can be included, thus allowing the creation of new filters and, consequently, visualize new results.

The data assessment is performed in the same parallel scheme as the simulation. Its objective is to build both two- and three-dimensional histograms. These histograms account for density, flux or integral number of neutrons distributed in any combination of selected dimensions. The histograms are calculated independently for each execution and then summed after all histograms are computed. Finally the data of each histogram is stored in a text file to be read by *gnuplot*.

In Figure 4.12 the graph of files included by the data process function is shown. This helps illustrate the classes required by this function.

In Figure 4.12, special attention is given to the histogram class. This is an abstract base class that is inherited by its two- and three-dimensional specialized classes, see Figure 4.13. It also holds methods for data output, as well as the methods that actually compute the histograms values. In these calculation a series of conditional functions are used, in order to count or disregard the particles, based on wether their tags respect certain conditions or not.

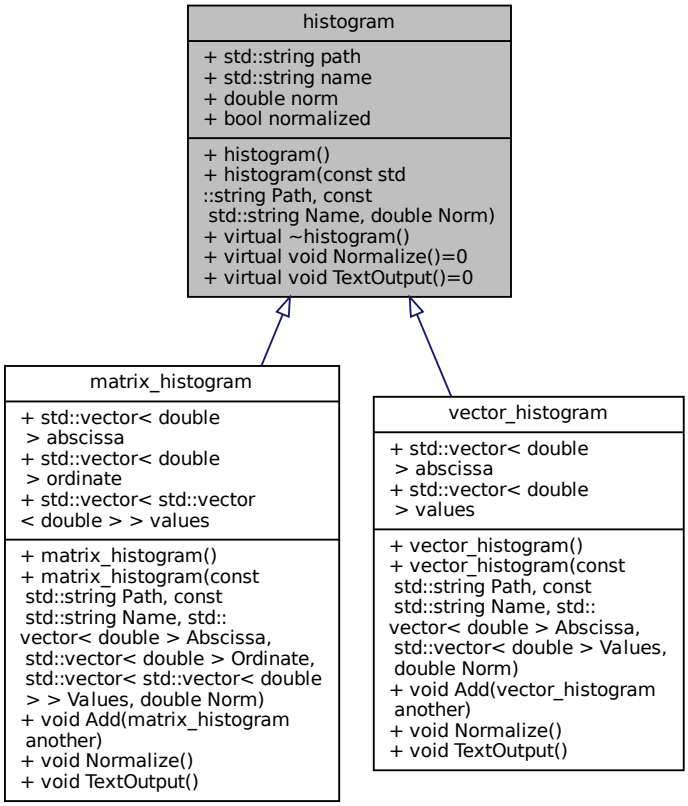


Figure 4.13 – Abstract histogram class inherited by its derived classes.

# 5 RESULTS

In order to demonstrate the simulator capabilities, four simulations were performed. The first two are a shielding simulation and a criticality benchmark simulation, and were published in Barcellos et al., 2019 and Chaves Barcellos et al., 2021, respectively. The third and fourth simulations comprehend a general scenario of a multiplicative medium, in order to compare both target at rest and free gas models, as well as using these results to asses the solution of the Boltzmann transport equation.

It is important to highlight that, in all simulations, geometry, chemical composition, and initial neutron population were changed. However, the program does not heave any kind of dedicated "shielding-mode" or "criticality-mode", and no further changes were required to run the different kinds of simulation, which are all genuine physical Monte Carlo procedures.

## 5.1 Shielding Simulation

This shielding simulation was published in Barcellos et al., 2019, as part of this thesis project, and its results are reproduced here. Its aim was to test the computation of reaction rates and mean free path, in a non-multiplicative medium.

### 5.1.1 Simulated Problem

A shielding problem was chosen to test the simulator. This is a well documented problem in literature, and an analytical solution can be approximated. These are time independent problems in a non multiplicative medium.

The shielding material used is cement, both standard and heavyweight cement are compared in these simulations. In Piotrowski et al., 2012 the atomic composition of both standard and heavyweight cement mixture are presented. Based on these references the atomic percentages of Table 5.1 were used in the simulations. For standard cement a density of $2\,g/cm^3$ and $2.6\,g/cm^3$ for heavyweight cement was considered.

In the simulation a semi-infinite medium was considered using a single material composed of an equivalent cement molecule in the proportions presented in Table 5.1, alongside the free gas scattering assumption. A mono-energetic beam of $1\,MeV$ entering orthogonally the semi-infinite medium was used.

Table 5.1 – Atomic percentages of simulated cement mixtures.

| Element | Atomic Percentage | |
|---|---|---|
| | Standard | Heavyweight |
| H | 0.795% | 0.622% |
| O | 53.724% | 31.758% |
| Al | 1.57% | 0.692% |
| Si | 38.808% | 2.96% |
| S | 0.131% | 10.666% |
| Ca | 4.298% | 3.904% |
| Fe | 0.674% | 4.124% |
| Ba | 0.000% | 45.274% |

A second case was simulated in which the scattering reaction was turned-off. This way only radiative capture occurred. Without the slowing down caused by the scattering, the neutron flux remains mono-energetic, and the solution for the flux that leaves a semi-infinite slab of non-multiplicative material of length $L$ is given by

$$\phi_L = \phi_0 \, e^{-L \, \Sigma_c}. \tag{5.1}$$

Where $\phi_0$ is the neutron flux entering the semi-infite slab, $\phi_L$ is the neutron flux leaving a slab of length $L$, and $\Sigma_c$ is the radiative capture macroscopic cross-section of the material for the energy of $1\,MeV$.

### 5.1.2 Results

Each case was computed with $10^6$ starting neutrons. Control volumes were then introduced and the flux leaving each volume was used to create histograms. In order to simulate different lengths for the slab, without the need for more than one execution, neutron tags were used so that the sub-volumes only accounted for neutrons entering from the side nearer to the source.

Figure 5.1 shows the number of counted neutrons by the slab length. It is noteworthy that the heavyweight cement acts as a neutron reflector for the simulated energy range, this is due to barium having a considerable scattering cross-section and being a heaviest nuclide of the two scenarios. After a few centimeters the effect of reflecting the neutrons back to the source begins to diminish, and the down-scattering followed by capture becomes dominant. This way, the counted neutrons in standard cement, with a

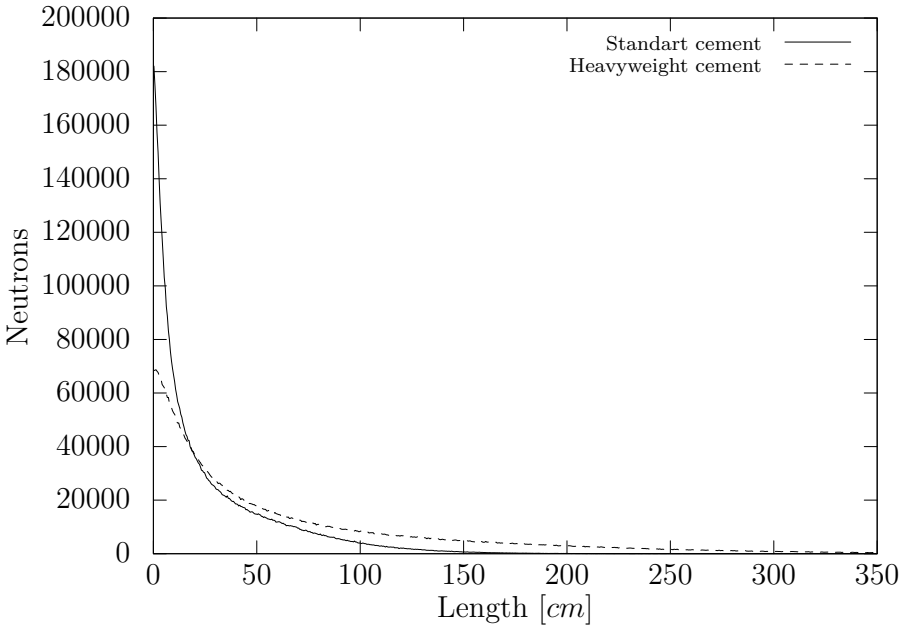


Figure 5.1 – Counted neutrons by slab length.

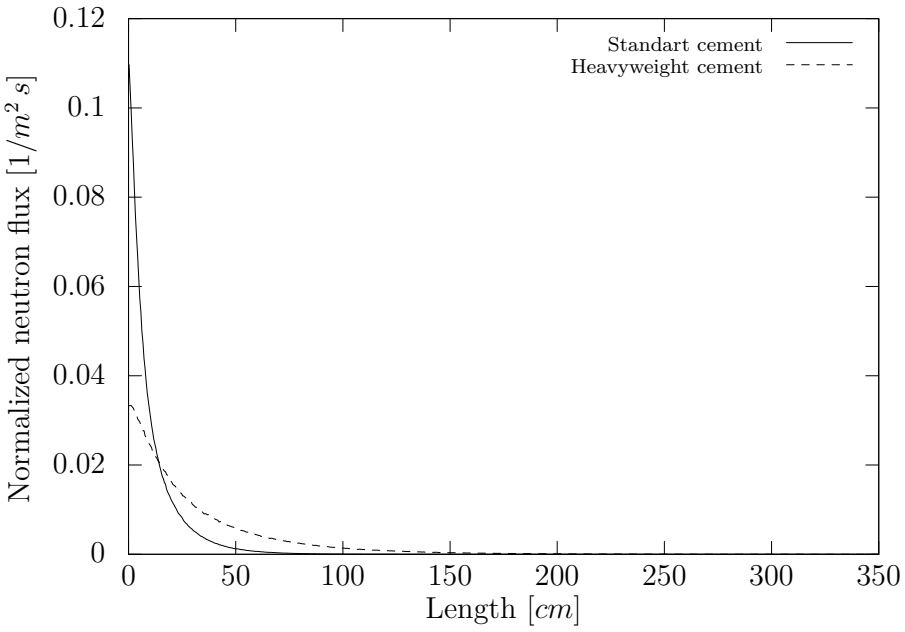


Figure 5.2 – Normalized neutron flux by slab length.

higher proportion of oxygen, becomes smaller.

In Figure 5.2 a similar effect can be seen for the neutron flux. However, this time the effects are more prominent. This is due to the fact the standard cement has, in average lighter nuclei, having thus a grater down-scattering capacity.

By switching off the scattering it is possible to compare the program result with the

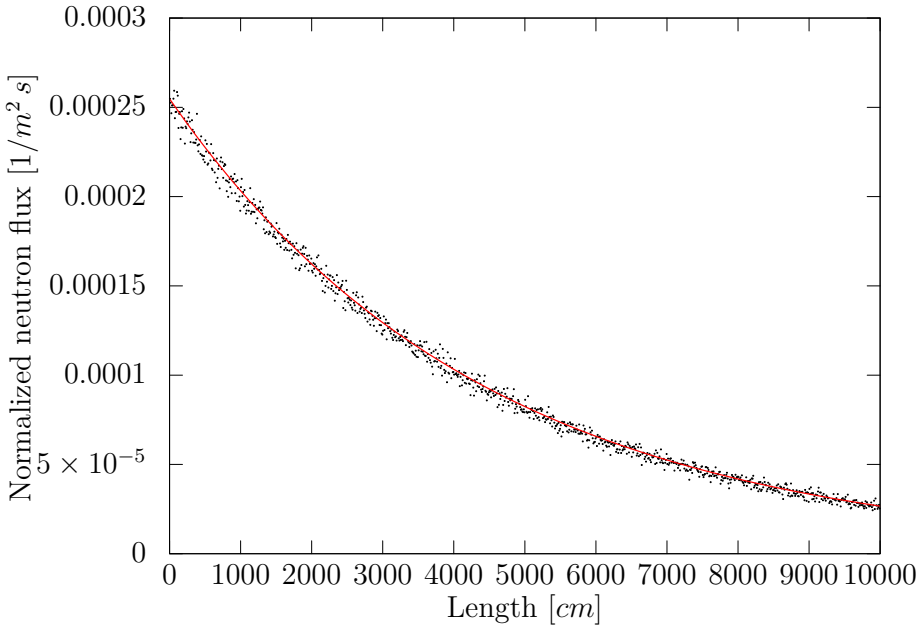


Figure 5.3 – Normalized neutron flux by slab length for standard cement in capture only medium.

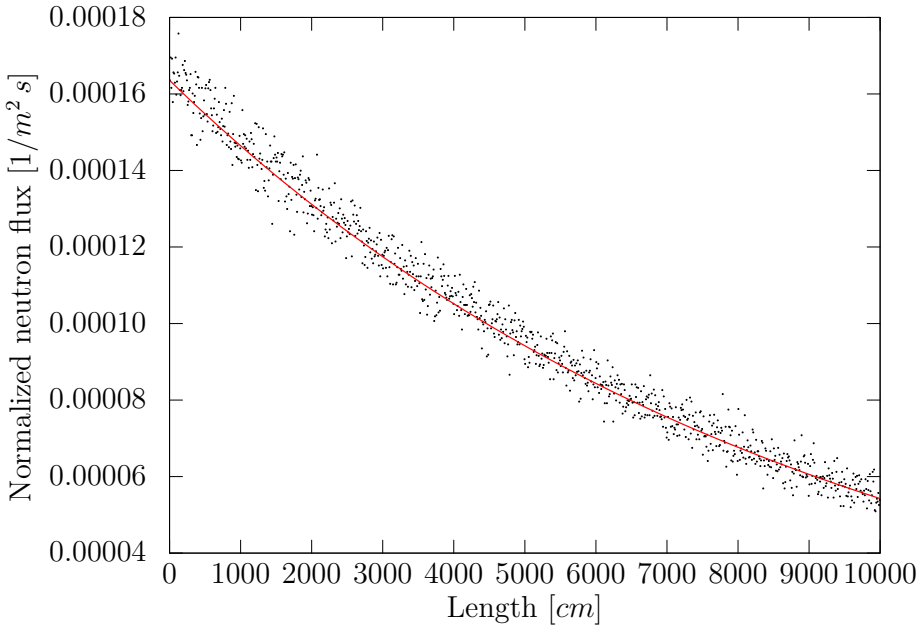


Figure 5.4 – Normalized neutron flux by slab length for heavyweight cement in capture only medium.

analytical solution in Equation 5.1. The comparison are shown for neutron flux in Figures 5.3 and 5.4. Even though this scenario does not represent a real case, it is important to ascertain the effect of down-scattering in shielding material. Nevertheless, here only the

radiative capture cross-sections for elements of the mixtures was considered, so that these results provide the pure capture mean free path for the neutrons, which is required for the comparison with Equation 5.1.

Table 5.2 – Computational statistics for $10^6$ initial neutrons.

| | Standard | Heavyweight | |
|---|---|---|---|
| Simulation thread-time | 1.40 | 1.76 | $[h \cdot thread]$ |
| Data assessment thread-time | 0.15 | 0.24 | $[h \cdot thread]$ |
| Maximum RAM per thread | 602.4 | 615.1 | $[MB/thread]$ |
| Stored simulation data | 4.5 | 5.5 | $[GB]$ |

Tables 5.2 and 5.3 show the computational statistics for all performed simulations. The great time reduction between the simulation considering scattering in Table 5.2 and the one with radiative capture only in Table 5.3 is plausible, for the simulator always ends a Monte Carlo step with a reaction, therefore, without considering scattering, all $10^6$ neutrons are captured within the first step.

Table 5.3 – Computational statistics for $10^6$ initial neutrons without scattering.

| | Standard | Heavyweight | |
|---|---|---|---|
| Simulation thread-time | 3 | 3 | $[min \cdot thread]$ |
| Data assessment thread-time | 1 | 1 | $[min \cdot thread]$ |
| Maximum RAM per thread | 289.5 | 287.8 | $[MB/thread]$ |
| Stored simulation data | 0.4 | 0.4 | $[GB]$ |

## 5.2 Criticality Simulation

This criticality simulation was published in Chaves Barcellos et al., 2021, as part of this thesis project, and its results are reproduced here. Its aim was to compare the result of a stationary problem to the results of other available Monte Carlo simulators.

### 5.2.1 Simulated Problem

An implementation that makes use of the created features is a criticality scenario from the International Handbook of Evaluated Criticality Safety Benchmark Experiments book [Pitts et al., 2018]. A Water-Reflected 91-Liter Sphere of Enriched Uranium Oxyfluoride Solution was chosen for this role due to its geometrical and chemical simplicity

and the fact that this configuration is considered acceptable for use as a criticality safety benchmark experiment [Pitts et al., 2018].

The experimental setup comprises three concentric spherical regions. The innermost region contains a solution of water and highly enriched uranium oxyfluoride, the second is a metallic shell that encloses the mixture and is composed of 1100 aluminum, and the third region is a water reflector pool that is assumed spherical. This assembly is summarized in Table 5.4, in which the outer radius of each region is also indicated.

Table 5.4 – Assembly summary.

| Region | Material | Outer radius |
|---|---|---|
| 1 | Water and uranium oxyfluoride solution | $27.9244\,cm$ |
| 2 | 1100 aluminum | $28.1244\,cm$ |
| 3 | Water | $43.1244\,cm$ |

The solution of the first region has a density of $1.0265\,g/cm^3$ and the isotopic weight fractions of uranium present in the $UO_2F_2$ molecule are given in Table 5.5. The experiment records also report that the uranium oxyfluoride molecule has a molecular weight of $305.1933\,a.m.u.$, however, using natural isotopic abundance for oxygen and fluor the computed molecular weight shall be $305.1843197\,a.m.u.$, which was used in the present simulation.

Table 5.5 – Uranium isotopic abundance.

| Isotope | Weight fraction |
|---|---|
| $^{234}U$ | 0.0098 |
| $^{235}U$ | 0.9318 |
| $^{236}U$ | 0.0050 |
| $^{238}U$ | 0.0534 |

For the aluminum shell, the 1100 aluminum is described as having a density of $2.71\,g/cm^3$. Its composition is given in Table 5.6, which defines the weight percentages of its constituent elements.

Finally, the water reflector has a density of $0.99705\,g/cm^3$. A global temperature of $298.15\,K$ is assumed for the three regions. Due to the fact that no information was given about the neutron spectrum, the present simulation started with a pure fission spectrum given by the Watt distribution

Table 5.6 – 1100 aluminum composition.

| Element | Weight fraction |
|---|---|
| $Al$ | 0.9870 |
| $Si$ | 0.0095 |
| $Cu$ | 0.0020 |
| $Mn$ | 0.0005 |
| $Zn$ | 0.0010 |

$$\chi(E) = 0.453\, e^{-1.036\ MeV^{-1}\, E} \sinh \sqrt{2.29\ MeV^{-1}\, E}. \tag{5.2}$$

According to the experiment records the effective multiplication factor determined was $k_{eff} = 0.9999$. Simulating the same setup with the codes ONEDANT (27-Group ENDF/B-IV) and XSDRNPM (238-Group ENDF/B-V) an estimated uncertainty of the benchmark is $\Delta_{k_{eff}} = \pm 0.0058$.

### 5.2.2 Results

The results obtained in this work were based on initially $10^5$ neutrons. Two independent simulations with initially $7.5 \times 10^4$ and $5 \times 10^4$ neutrons were executed with different random seeds in order to analyse the stability of the spectral flux as well as the effective multiplication factor. The initial neutrons from fission were distributed uniformly inside the uranium oxyfluoride solution. The simulations segments were limited to 10000 Monte Carlo steps, during which 203 ($10^5$ initial neutrons), 202 ($7.5 \times 10^4$ initial neutrons), and 201 ($5 \times 10^4$ initial neutrons) neutron generations were identified. Due to the spectral bias introduced by this initial population by fission neutrons, the first 750 Monte Carlo steps with its 5 neutron generations were ignored for the evaluation of the results.

Recalling, that from one and the same simulation for each respective initial neutron population all simulation data was stored, so that the spectral flux and $k_{eff}$ could be determined. The spectral and spatial distributions were shown only for the simulation with the largest number of initial neutrons.

#### 5.2.2.1 Neutron Density and Neutron Flux

In addition to the information of the effective multiplication factor, the data saved during simulation can be used, among others, to investigate the behaviour of the neutron flux and neutron density. The neutron density by radial position is shown in Figure 5.5, with dashed lines to define the radius of each region.

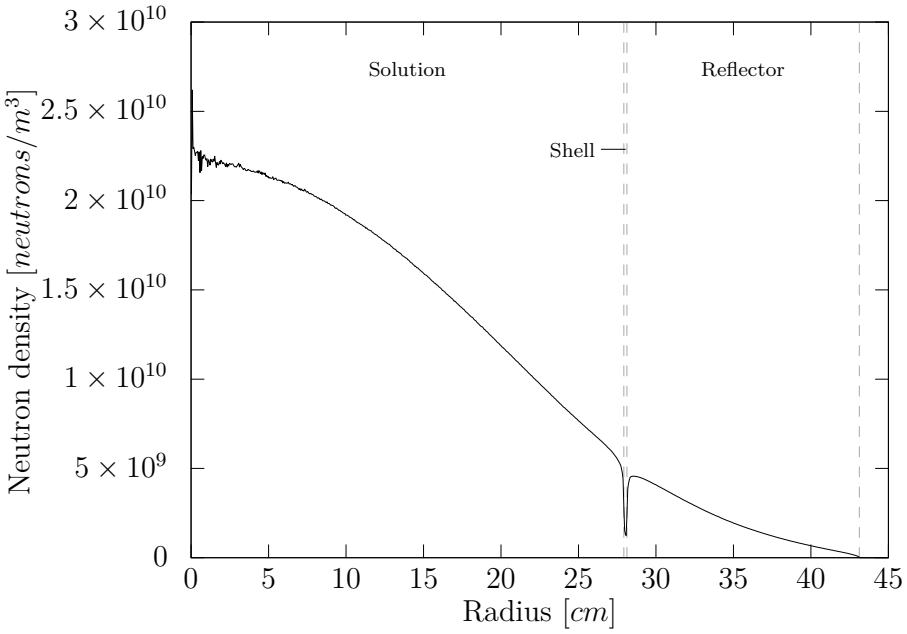


Figure 5.5 – Neutron density by radial position.

Also the neutron flux by radial position is shown in Figure 5.6, again with dashed lines to define the radius of each region. An interpolation of the spectral neutron flux by a single continuous analytical function was performed and showed alongside the simulation results.

It is noteworthy that both flux and density at the boundary of simulation are not null. This is worth mentioning, since no assumption was made regarding the boundary condition and, in fact, this result can be used to feed an analytical or semi-analytical approach usually based on a diffusion model, where typically assumptions for the boundary are necessary in order to obtain an unique solution. These type of approaches also do not treat energy as a continuous variable but rather integrate energy intervals into groups. Thus, the constituting equation system needs the averaged cross sections, where the flux solution is necessary, in principle, to get these values accurately.

The simulation provides the spectral neutron flux shown in Figure 5.7 which may

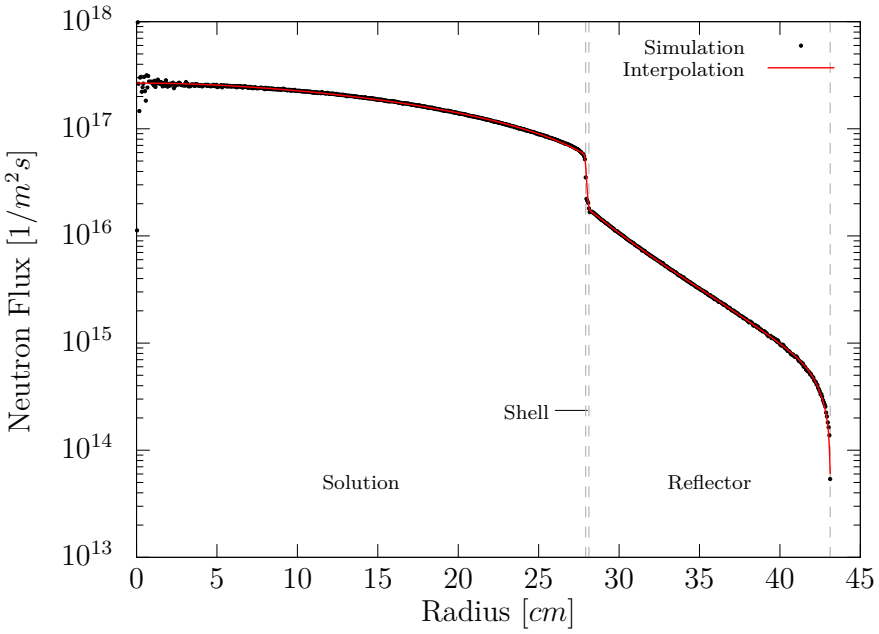


Figure 5.6 – Neutron flux by radial position.

be used to define accurate values for the group calculations. In this figure the spectral flux of each region is shown normalized by the spectral flux in the whole domain. The escape spectral flux is also shown and, for the sake of comparison, it is also normalized according to the integral of the spectral neutron flux.

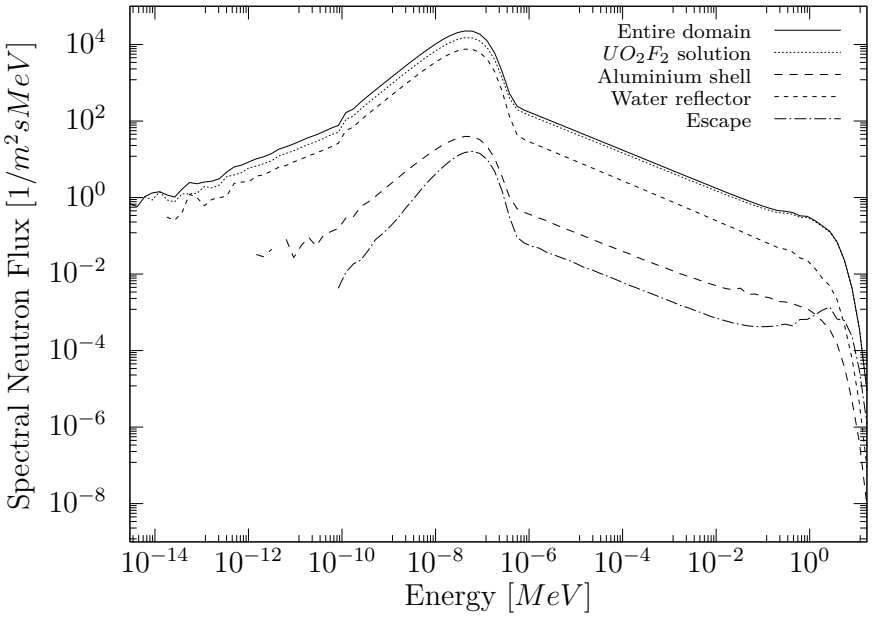


Figure 5.7 – Spectral neutron flux.

#### 5.2.2.2 Computational Statistics

The computational details of the simulation procedure is presented in Table 5.7. In this table the simulation time comprehends only the time required to compute and store the data in binary files, whereas the data assessment time is the computational time to read these binary files and compute the desired histograms.

Table 5.7 – Computational details for $10^5$ initial neutrons simulated for $10^4$ Monte Carlo steps.

| | | |
|---|---|---|
| Simulation thread-time | 8.57 | $[h \cdot thread]$ |
| Data assessment thread-time | 1.05 | $[h \cdot thread]$ |
| Maximum RAM per thread | 52.6 | $[MB/thread]$ |
| Stored simulation data | 228.8 | $[GB]$ |

#### 5.2.2.3 Effective Multiplication Factor

Identification of the neutrons that belong to a specific generation is necessary to determine $k_{eff}$. The effective multiplication factor is computed by the ratio of the number of neutrons born from fission in a given generation to the number of neutrons born from fission in the previous one. In the present implementation the neutrons get a tag that identifies the cycle they belong to. The effective multiplication factor determined from simulation is shown in Figure 5.8 alongside the ratio of neutrons of each generation that were still under tracking when the simulation was halted at step 10000 (in bars). Note that these neutrons remain unaccounted until they finish their tracking in either an absorption interaction, or an escape. Neutron generations are not synchronized during simulation, and neutrons whose tracking is still being processed during simulation termination could alter the computation of $k_{eff}$. Thus the generations of these neutrons cannot be used to calculate $k_{eff}$ since the number of fissions to be compared to the number of fissions in the previous generation is no longer representative. The generations used to determine $k_{eff}$ as well as the incomplete cycles that no longer are taken into account are indicated in Figure 5.8.

More specifically, the generations in the range from 5 to 127 were used to compute the mean effective multiplication factor. The first generations were excluded due

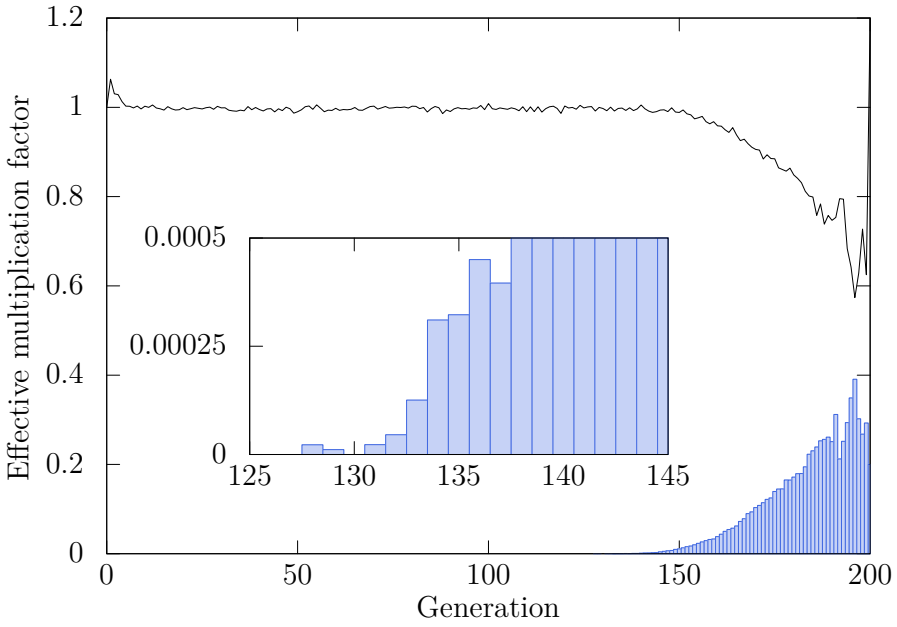


Figure 5.8 – Effective multiplication factor $k_{eff}$ by generation and unaccounted part of the simulation.

to the bias from the initial fission spectrum population. The latest generations were excluded for they were not entirely sampled and are due to the simulation termination. The mean effective multiplication factor from the interval of 123 generations was calculated to $k_{eff} = 0.99732$. The standard deviation of this sample was also computed and is $\sigma_{K_{eff}} = \pm 0.00419$ of the same magnitude as the uncertainty of the benchmark experiment.

In Figure 5.9 a detail for the $k_{eff}$ in the generation interval [5, 127] is shown. The figure also shows the mean $k_{eff}$ in this interval with its $1\,\sigma_{k_{eff}}$ and $3\,\sigma_{k_{eff}}$ intervals, as well as the result from the benchmark experiment $k_{eff} = 0.9999$. Note that the measured value is within $1\,\sigma_{k_{eff}}$ from the simulated average value.

The article on the benchmark experiment [Pitts et al., 2018] also contains statistics with results obtained with other simulators. These tables are reproduced and summarized in Table 5.8. The results for the $k_{eff}$ of independent simulations initialized with $7.5 \times 10^4$, and $5 \times 10^4$ fission neutrons are added to this table. Note that their mean effective multiplication factors were computed in the generation intervals of [5, 116] and [5, 131], respectively, following the same criterion as in the case of initially $10^5$ neutrons. Table 5.8 also contains the relative difference $(C - E)/E$, in which $C$ represents the calculated value and $E$ the experimental value, normalized by the experimental result of 0.9999.

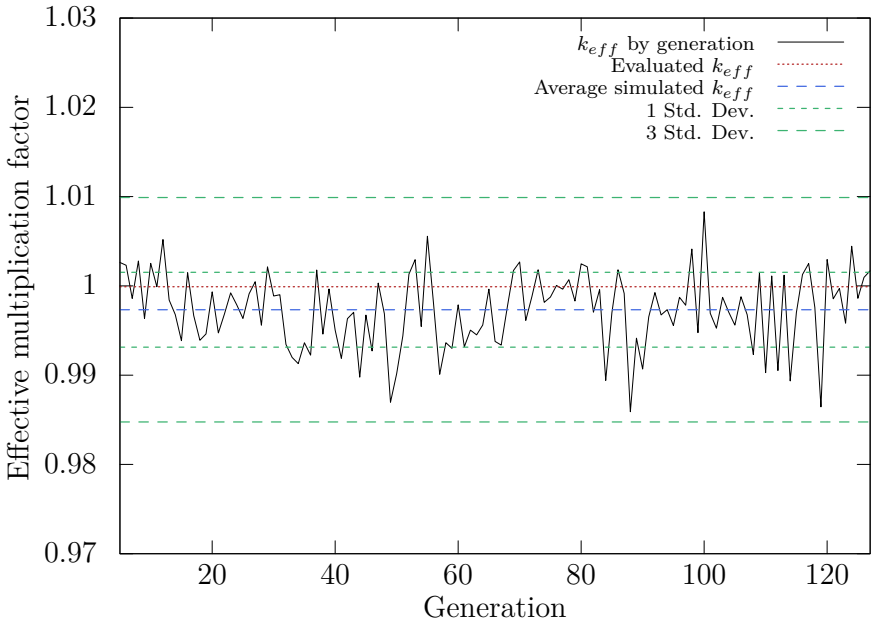


Figure 5.9 – Effective multiplication factor by generation and unaccounted part of the generation.

Table 5.8 – Summary of results for the benchmark simulation.

| Results | $k_{eff} \pm \Delta k_{eff}$ | Relative Difference |
|---|---|---|
| Benchmark[1] | $0.9999 \pm 0.0058$ | $0 \pm 0.58\%$ |
| Present work ($10^5$ starting neutrons) | $0.9973 \pm 0.0042$ | $-0.26\% \pm 0.42\%$ |
| Present work ($7.5 \times 10^4$ starting neutrons) | $0.9980 \pm 0.0043$ | $-0.19\% \pm 0.43\%$ |
| Present work ($5 \times 10^4$ starting neutrons) | $0.9981 \pm 0.0053$ | $-0.18\% \pm 0.53\%$ |
| KENO-Va (Hansen-Roach)[1] | $1.0072 \pm 0.0016$ | $0.73\% \pm 0.16\%$ |
| KENO-Va (27-Group ENDF/B-IV)[1] | $1.0013 \pm 0.0012$ | $0.14\% \pm 0.12\%$ |
| KENO-Va (299-Group ABBN-93)[1] | $1.0011 \pm 0.0002$ | $0.12\% \pm 0.02\%$ |
| MCNP4 (Continuous-Energy ENDF/B-V)[1] | $1.0037 \pm 0.0005$ | $0.38\% \pm 0.05\%$ |
| MCNP5 (ENDF/B-VI.8)[1] | $1.0015 \pm 0.0001$ | $0.16\% \pm 0.01\%$ |
| MCNP5 (ENDF/B-VII $\beta 3$)[1] | $1.0009 \pm 0.0001$ | $0.10\% \pm 0.01\%$ |
| ONEDANT (27-Group ENDF/B-IV)[1] | 1.0037 | 0.38% |
| XSDRNPM (238-Group ENDF/B-V)[1] | 1.0023 | 0.24% |
| MONK7A (Continuous-Energy JEF2.2)[1] | $1.0006 \pm 0.0010$ | $0.07\% \pm 0.10\%$ |

It is noteworthy that, although the KENO as well as the MCNP simulations are indicated with uncertainties of the order of magnitude smaller than the benchmark experiment result or our simulations, the three results obtained with the KENO code have no common interval on the one $\sigma$ level and the same is true for the results simulated with MCNP. By inspection of Table 5.8 one also observes that our simulations are the only

[1]Extracted from Pitts et al., 2018.

ones with effective multiplication factors smaller than unity, all the other nine simulations provide multiplication factors between 1.0006 and 1.00072 and in most cases unity is even excluded by the indicated uncertainties.

Moreover, MCNP uses the power iteration method [Brown, 2005] to compute $k_{eff}$, and for the results presented in Table 5.8 2500 generations were run, each with 2000 neutrons, being the first 150 generations discarded [Pitts et al., 2018]. In comparison, the results of this paper were attained by averaging the $k_{eff}$ of 111, 121 and 126 generations, in which the total neutron number varies, but encompasses the totality of the population.

## 5.3 Spectral Neutron Flux

In order to compare both fixed target and free gas models, a scenario of a homogeneous multiplicative medium was created. The simulation was run with 25000 starting fission neutrons for a total of 3000 Monte Carlo steps. Two runs were performed, one assuming the fixed target hypothesis, and the other using the free gas model.

The simulated scenario comprised of a $1\,m^3$ cube, containing a homogeneous mixture of 72 % volume water and 28 % volume uranium dioxide. In which the uranium has an enrichment of 0.895 %.

In Figure 5.10 we can see the comparison between the spectral neutron flux of each model. In this figure we perceive that the neutron flux of the fixed target model has a tendency towards lower energies, in relation to the free gas model.

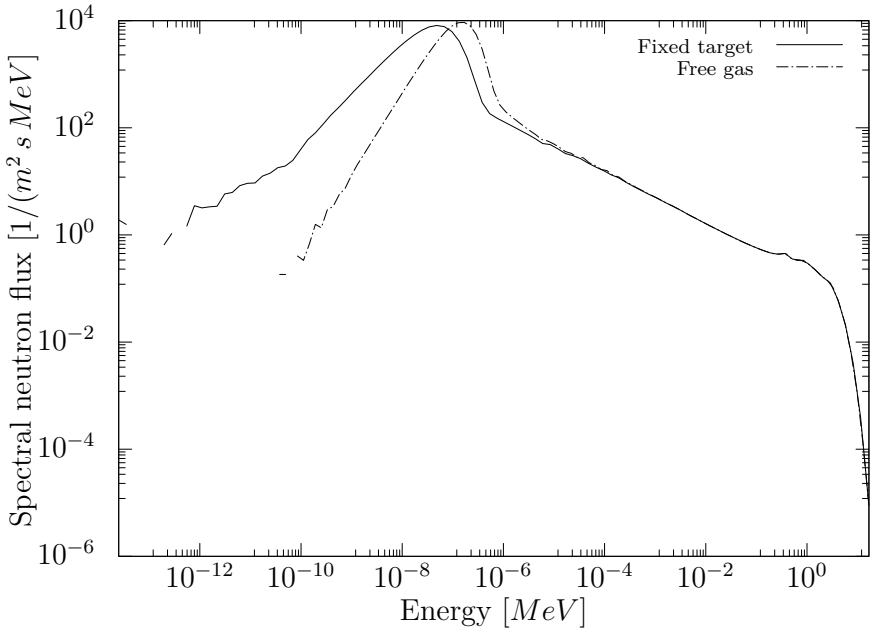


Figure 5.10 – Neutron flux spectra of fixed target and free gas models.

### 5.3.1 Effective Multiplication Factor

These differences in neutron flux spectra result in a difference in the effective multiplication factor, i.e. the ratio between fissions induced by neutrons of a given generation to the number of fissions caused by neutrons of the previous generation. This is shown in Figure 5.11, with results for both models.

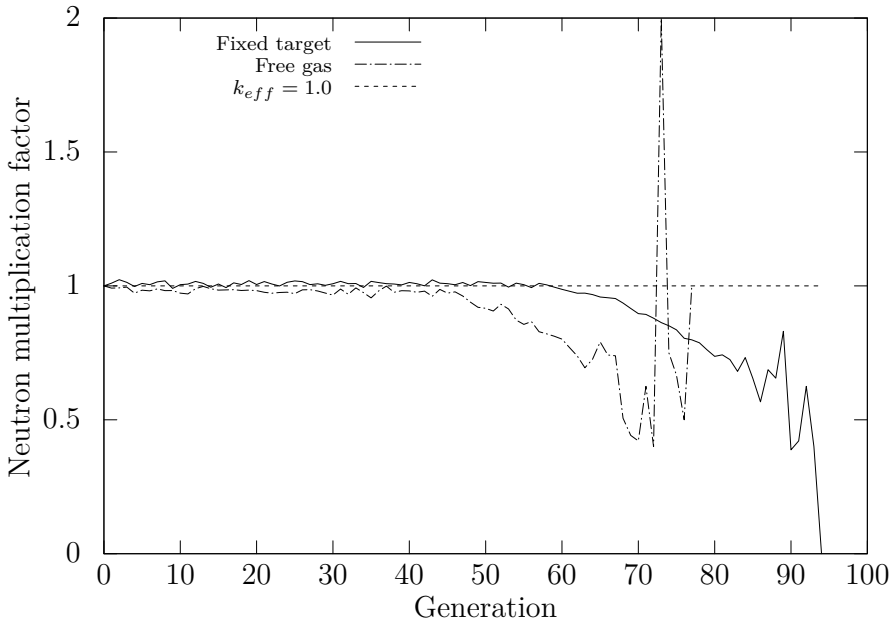


Figure 5.11 – Effective multiplication factor by generation.

It is possible to note that for the later generations, the neutron multiplication factor deviates from its previous behavior. This can be explained by the fact that these later generations are not fully sampled. As the simulations are terminated at step 3000, the neutrons of these later generations have their tracking stopped, and do not count for either fission, capture, or escape.

### 5.3.2 Time Distributions

Due to the difference between neutron flux spectra, the correlation between Monte Carlo step and time, is also changed. As stated before, a Monte Carlo step does not correlate to a fixed time interval, but to a distribution. This relation between Monte Carlo step and time can be seen in Figure 5.12 for the target at rest model, and in Figure 5.13 for the free gas model.

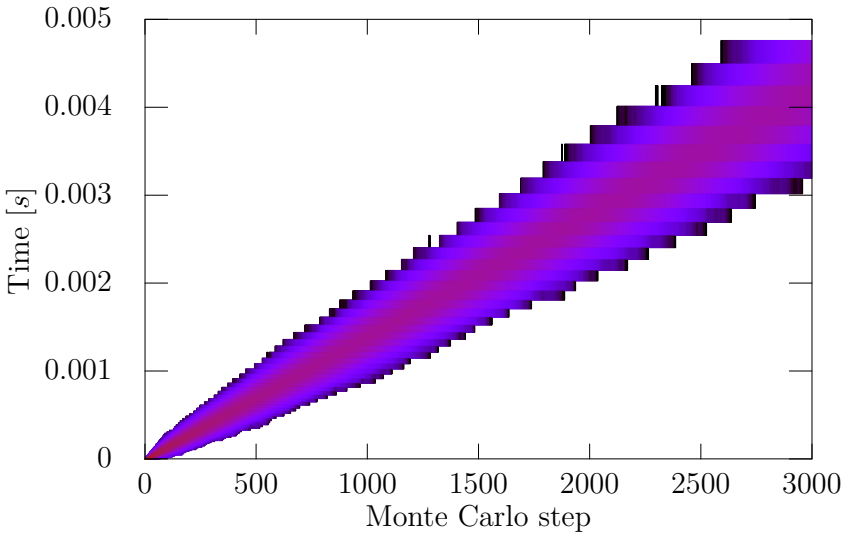


Figure 5.12 – Monte Carlo step time distribution for the target at rest model.

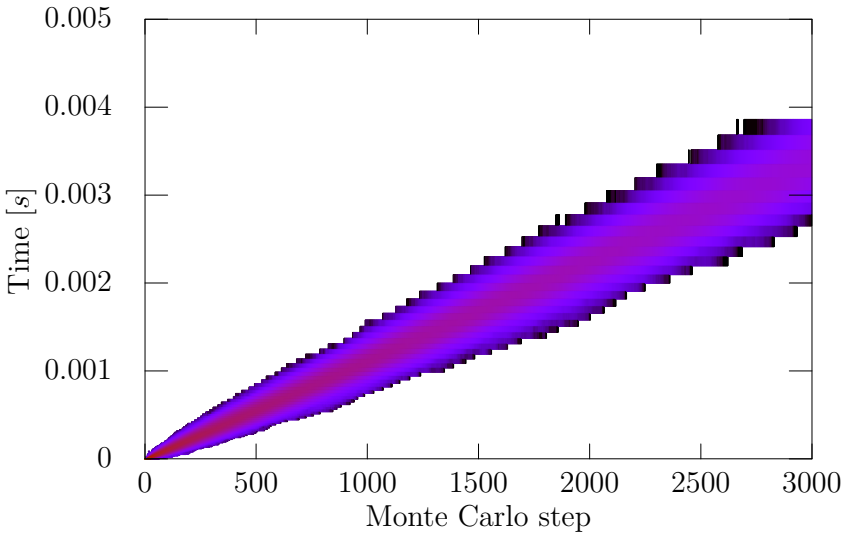


Figure 5.13 – Monte Carlo step time distribution for the free gas model.

In Figures 5.12 and 5.13 it can be seen that the mean step time presents a linear correlation to the Monte Carlo step. The standard deviation, however, continuously increases with the simulation steps. Also, the mean step time of the free gas model is smaller than the one of the target at rest, what was expected due to the free gas model presenting a higher mean energy.

In order to find a recurrent regime, and thus be able to find a stationary neutron flux, the evolution of the neutron flux in time must be evaluated. Two figures are shown for each model, Figures 5.14 and 5.16 show the behavior of the spectral neutron flux along all simulation period, while Figures 5.15 and 5.17 show the details of the first $10^{-3}$ $s$.

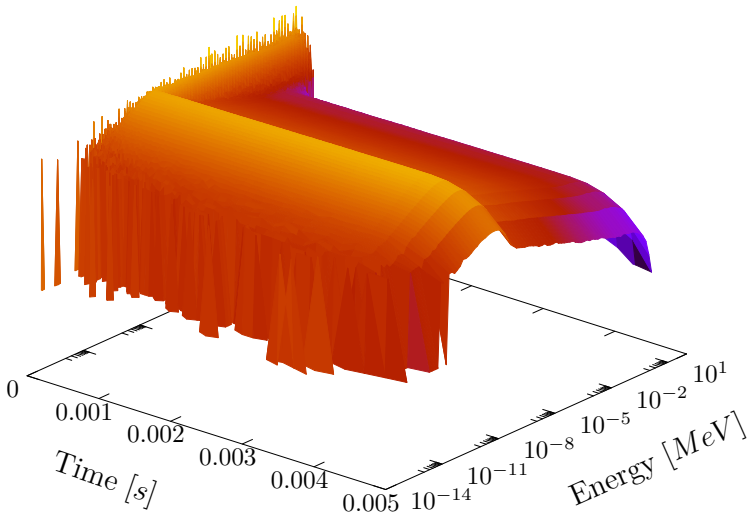


Figure 5.14 – Spectral neutron flux time evolution for the target at rest model.

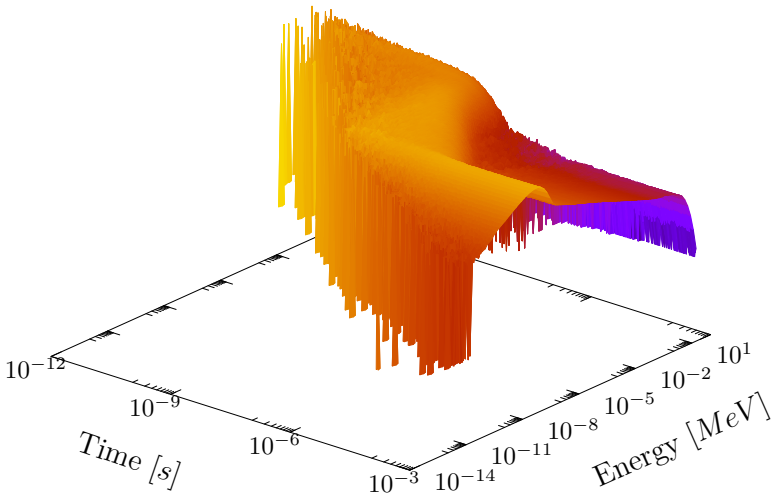


Figure 5.15 – Detail of the spectral neutron flux time evolution for the target at rest model.

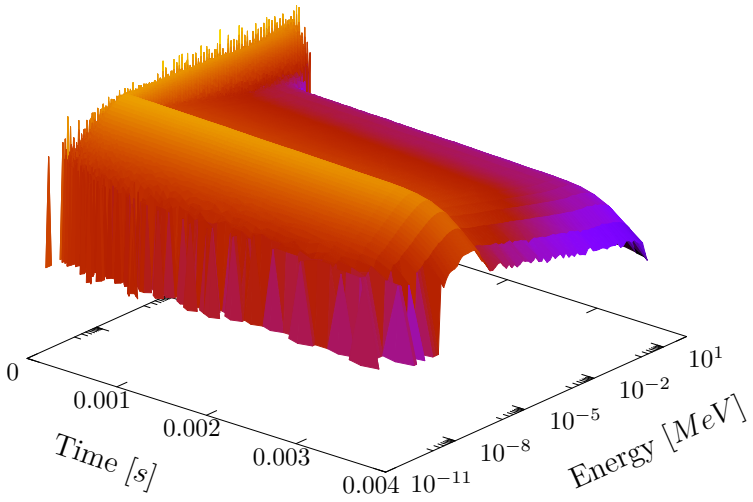


Figure 5.16 – Spectral neutron flux time evolution for the free gas model.

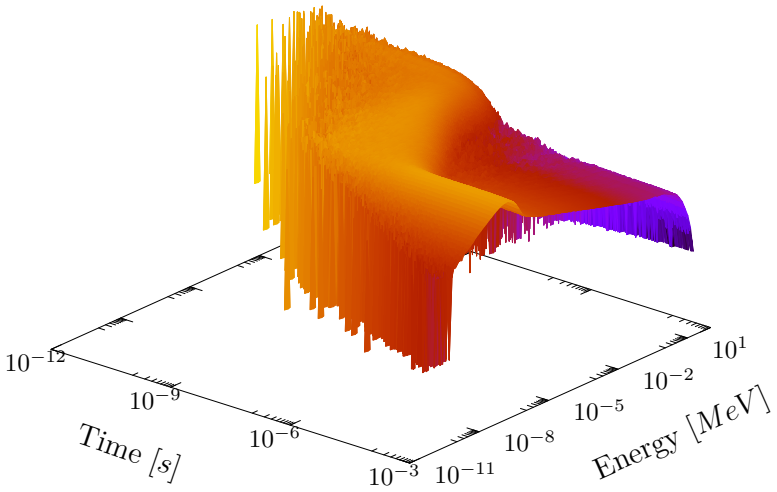


Figure 5.17 – Detail of the spectral neutron flux time evolution for the free gas model.

In Figures 5.14 and 5.16 it is possible to perceive that the neutron flux spectrum remains constant, within a small statistical oscillation, for the vast majority of steps. In Figures 5.15 and 5.17 the rapid transient of the first $10^{-3}$ $s$ is shown. Here it is clear that there is a change in the shape of the neutron flux. This shows the evolution of a purely fission population that follows the Watt spectrum to one that has attained stationarity.

### 5.3.3 Population by Distribution

Another result that allows to visualize how the three probability distributions relate to each other is the normalized number of neutrons in each distribution. Figures 5.18 and 5.19 show the normalized population of each distribution and how they add up to the total neutron population, for the fixed target model and free gas model, respectively.

It is noteworthy that there is no fixed energy value between the distributions, but a stochastic criteria that allows their superposition. This classification criteria can be applied to both scattering models, however, since for the free gas model it does not change the treatment of the neutron during tracking and interaction computation (as performed in the fixed target model), this classification is not used in any calculation of the simulator.

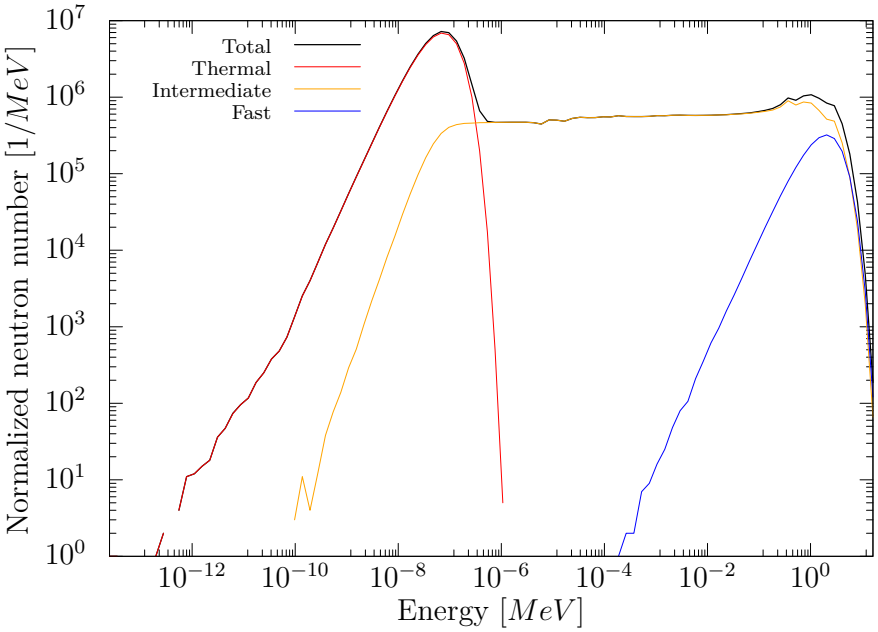


Figure 5.18 – Spectral neutron distribution and the contribution of each population for the target at rest model.

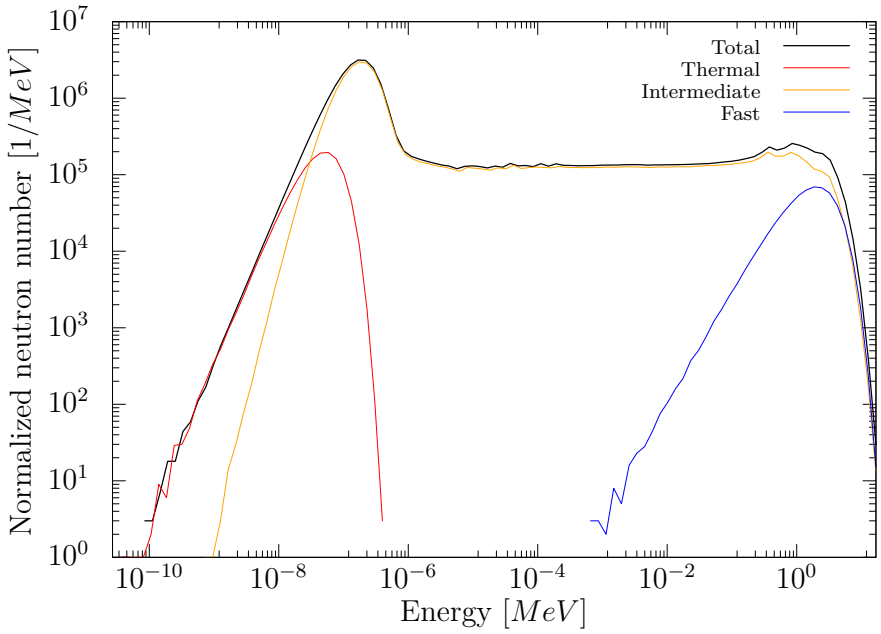


Figure 5.19 – Spectral neutron distribution and the contribution of each population for the free gas model.

## 5.4 Boltzmann Equation Evaluation

Both neutron flux spectra with their respective interpolating functions are presented in Figures 5.20 and 5.21. Each interpolating function is composed of seven rational functions of the type

$$\frac{a}{1+(\frac{x-b}{c})^2},$$

in which $x$ is the independent variable and $a$, $b$ and $c$ are the fitting coefficients, and the interpolation is performed in the log-log scale.

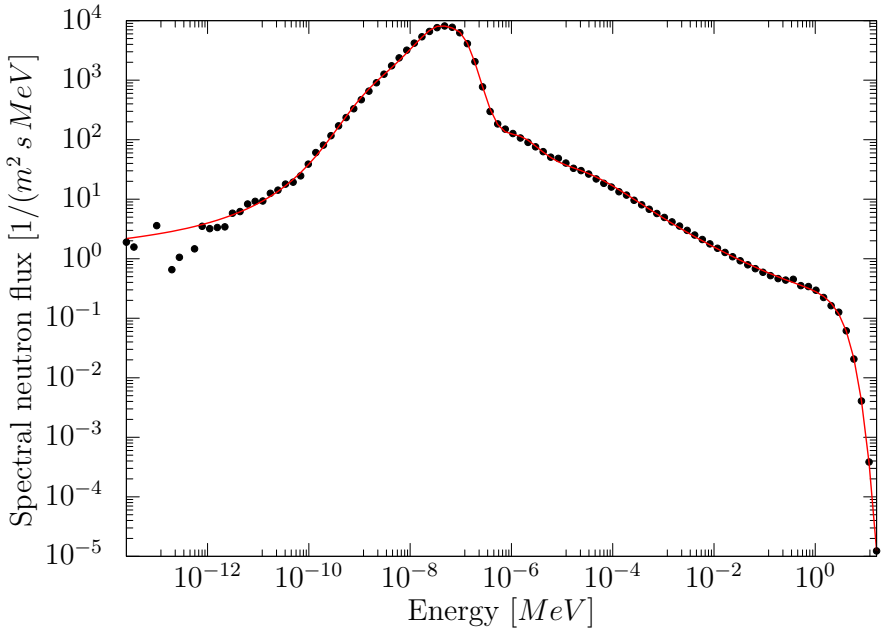


Figure 5.20 – Spectral neutron flux and interpolating function for the target at rest model.

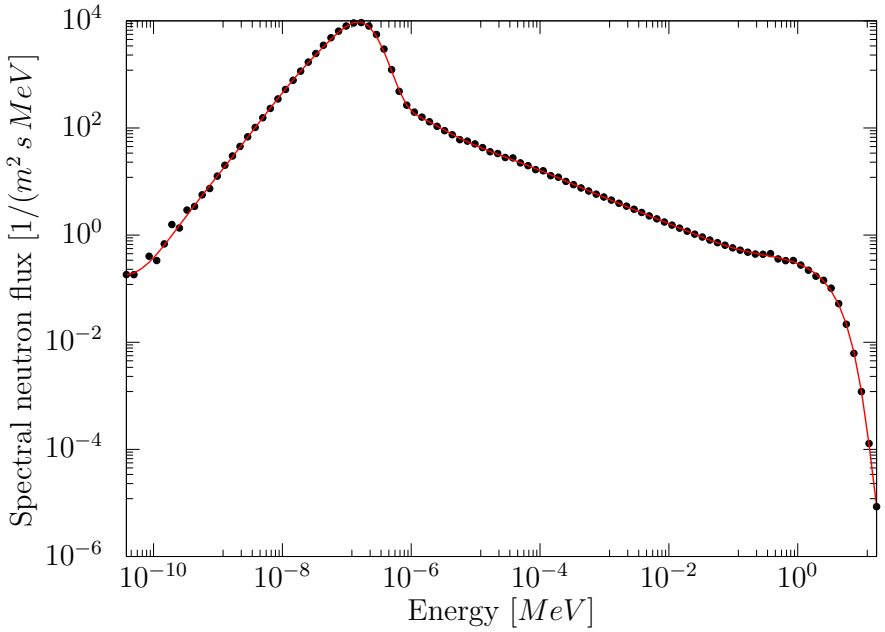


Figure 5.21 – Spectral neutron flux and interpolating function for the free gas model.

With the interpolating functions for the neutron flux spectra, it is possible to assess if they are good solution candidates of Equation 2.8. In this assessment the differential scattering probability $P_{i,s}(E' \rightarrow E)$ is computed numerically, as is shown in Figures 4.8

to 4.11.

A comparison between right and left hand sides of this equation is presented in Figures 5.22 and 5.23, for the target at rest and free gas models, respectively. In these figures the left hand side of Equation 2.8 is shown in blue and represents the removal of neutrons from the flux, while the right hand side represents the emissions of neutrons into the flux and is shown in red.

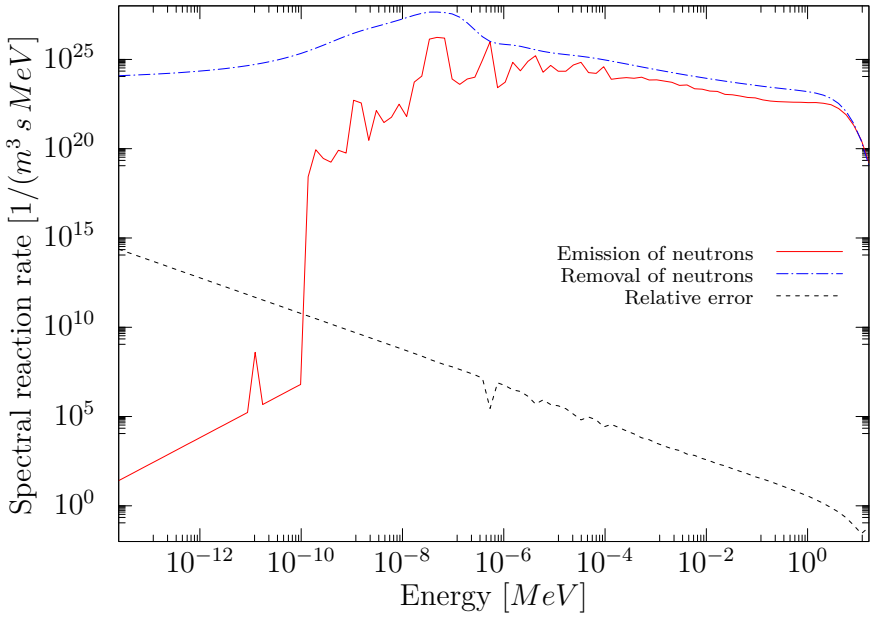


Figure 5.22 – Evaluation of a possible Boltzmann solution for the target at rest model.

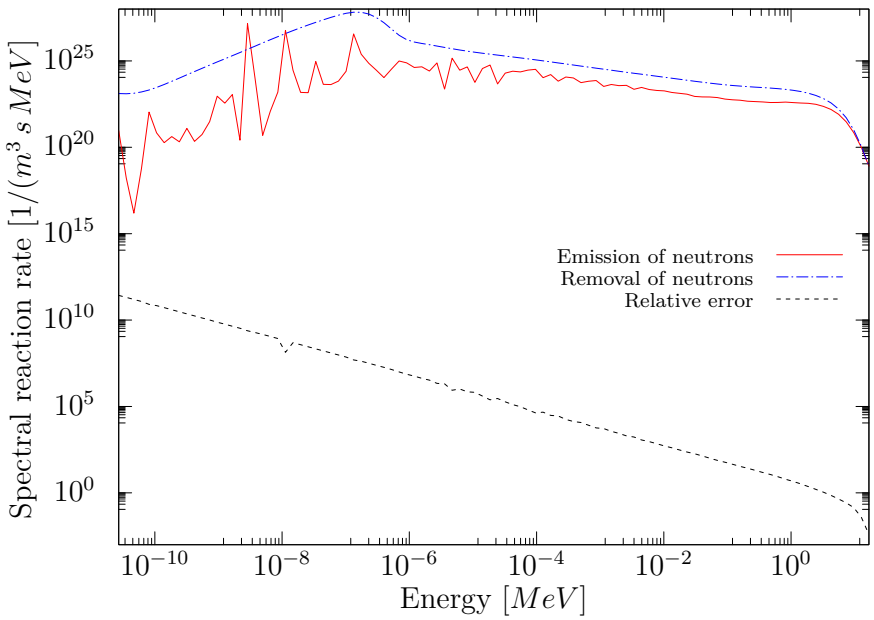


Figure 5.23 – Evaluation of a possible Boltzmann solution for the free gas model.

Due to the fact that physical Monte Carlo simulation uses simplifications, the pertinent question arises whether to use a fixed target or a free gas approach. In order to validate the quality of the aforementioned philosophies the found distributions were fed into the Boltzmann transport equation. By inspection of Figures 5.22 and 5.23 one observes smaller deviations in the higher energy regions for both models, whereas in the lower energy region, the fixed target model shows a profound discrepancy in the neutron balance. Thus, the free gas model approach, so far, is clearly the better option for neutron transport simulations.

# 6 CONCLUSIONS AND FUTURE WORK

The presented results showcase the developed program. They show that the Monte Carlo simulator configures a valid tool in the evaluation of the neutron population in a given medium. It shows itself versatile in the different kind of results that can be obtained, e.g. neutron flux, neutron density, reaction rates, criticality, among others. Even tough the simulator is continuous in all seven parameters of the phase space, it allows for the accumulation in the desired axes, integrating out desired dimensions, as was done in the results with time, space, azimuthal and/or polar angles.

A shielding scenario, a criticality scenario, and two simulations in a multiplicative medium, each with a different scattering model, were simulated. Several types of results were obtained from these simulations, among these are results for neutron density, effective multiplication factor, Monte Carlo step time distribution, spectral neutron flux, and last, but not least, the assessment of the fitness of the spectral neutron flux to the Boltzmann transport equation solution.

In the shielding simulation. Two different cement mixtures were used in the process, and two scenarios were computed, one for the problem including scattering and radiative capture, and one idealized scenario in which only radiative capture was considered. The idealized scenario had a simple analytical solution, which showed good agreement with the stochastic data.

The criticality simulation consisted of a benchmark from the International Handbook of Evaluated Criticality Safety Benchmark Experiments book. Here the results for effective multiplication factor was compared to other simulators, and showed good agreement to the measured results, being the only simulator that correctly resulted in a subcritical scenario. The neutron flux and neutron density by radial position were shown also shown, as well as the presentation of the spectral neutron flux for the inner regions and leakage. These results are of paramount relevance for they allow to use the simulation to find boundary conditions for analytical and semi-analytical methods.

Finally two simulations of a multiplicative medium are done with two different scattering models. For each model the spectral neutron flux, effective multiplication factor, and time distributions for the Monte Carlo step are shown. Also, the division of the neutron population in three separate distributions, that is performed by the simulator,

is presented. Lastly the spectral neutron fluxes are interpolated and these functions are verified as possible solutions of the Boltzmann transport equation.

As the balance between both sides of the Boltzmann transport equation shows, there are differences that arise from the simplifications of the chosen scattering models. These make necessary the use of more complete models, such as the use of angular distributions, parametrized from experimental data. Such upgrades to the code are already under development, and the use of object oriented programming makes these changes easy to implement, since there is no alteration in functionality, but only the expansion of interaction libraries.

The work developed in this thesis represents a milestone of a larger project. The simulator had its origin in de Camargo, 2011, but it was during the author's master degree and subsequent doctorate that the program was rewritten in modern C++ according to the latest parallelization paradigms of power computing. In these six years three conference papers, two book chapters and two articles were published [Barcellos et al., 2015, 2017a,b, 2019, 2020, 2021; Chaves Barcellos et al., 2021].

The project has further improvements already in progress. Some of these include the parametrization and further consideration of transient behaviours, molecular and solid state conditioned cross sections, the implementation thermal scattering with molecules and phonons, local temperature effects, delayed neutrons, among others. Since, physical aspects are implemented separately from computational ones, the program structure allows for reasonably easy maintenance and upgrades, according to new computational developments and physical insights.

Finally, the release of this code with an open license is planned. This will allow in the future integrating the developed code as a physics patch of GEANT. The author reasons, that with the satisfactory results that were obtained and the program's release, the open source nature and international collaboration will accelerate its future developments.

## BIBLIOGRAPHICAL REFERENCES

Agostinelli, S., Allison, J., Amako, K. a., Apostolakis, J., Araujo, H., Arce, P., Asai, M., Axen, D., Banerjee, S., Barrand, G., et al. Geant4a simulation toolkit, **Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment**, vol. 506(3), p. 250–303, 2003.

Barcellos, L. F. F., Bodmann, B. E., Leite, S. Q. B., and de Vilhena, M. T. **A Monte Carlo Simulation of a Simplified Reactor by Decomposition of the Neutron Spectrum into Fission, Intermediate and Thermal Distributions**. In International Nuclear Atlantic Conference 2015, 2015.

Barcellos, L. F. F., Bodmann, B. E., Leite, S. Q. B., and de Vilhena, M. T. **A Continuous Energy Neutron Transport Monte Carlo Simulator Project: Decomposition of the Neutron Energy Spectrum by Target Nuclei Tagging**. In International Nuclear Atlantic Conference 2017, 2017a.

Barcellos, L. F. F., Bodmann, B. E., Leite, S. Q. B., and de Vilhena, M. T. **On a Continuous Monte Carlo Simulator Project: Neutron Shielding Simulation**. In International Nuclear Atlantic Conference 2019, 2019.

Barcellos, L. F. F. C., Bodmann, B. E. J., de Vilhena, M. T. M. B., et al. A Continuous Energy Neutron Transport Monte Carlo Simulator Project: Decomposition of the Neutron Energy Spectrum by Target Nuclei Tagging, **Brazilian Journal of Radiation Sciences**, vol. 8(3B), 2021.

Barcellos, L. F. F. C., Bodmann, B. E. J., Leite, S. Q. B., and Vilhena, M. T., **On a Continuous Energy Monte Carlo Simulator for Neutron Transport: Optimisation with Fission, Intermediate and Thermal Distributions**, p. 1–10. Springer International Publishing, Cham, 2017b.

Barcellos, L. F. F. C., Bodmann, B. E. J., and Vilhena, M. T., **On a Parametric Representation of the Angular Neutron Flux in the Energy Range from 1 eV to 10 MeV**, chapter 3, p. 45 – 59. Springer Nature Switzerland AG, 2020.

Bell, G. and Glasstone, S. **Nuclear Reactor Theory**. Van Nostrand Reinhold Company, 1970.

Brown, D., Chadwick, M., Capote, R., Kahler, A., Trkov, A., Herman, M., Sonzogni, A., Danon, Y., Carlson, A., Dunn, M., Smith, D., Hale, G., Arbanas, G., Arcilla, R., Bates, C., Beck, B., Becker, B., Brown, F., Casperson, R., Conlin, J., Cullen, D., Descalle, M.-A., Firestone, R., Gaines, T., Guber, K., Hawari, A., Holmes, J., Johnson, T., Kawano, T., Kiedrowski, B., Koning, A., Kopecky, S., Leal, L., Lestone, J., Lubitz, C., Damián, J. M., Mattoon, C., McCutchan, E., Mughabghab, S., Navratil, P., Neudecker, D., Nobre, G., Noguere, G., Paris, M., Pigni, M., Plompen, A., Pritychenko, B., Pronyaev, V., Roubtsov, D., Rochman, D., Romano, P., Schillebeeckx, P., Simakov, S., Sin, M., Sirakov, I., Sleaford, B., Sobes, V., Soukhovitskii, E., Stetcu, I., Talou, P., Thompson, I., van der Marck, S., Welser-Sherrill, L., Wiarda, D., White, M., Wormald,

J., Wright, R., Zerkle, M., erovnik, G., and Zhu, Y. ENDF/B-VIII.0: The 8th Major Release of the Nuclear Reaction Data Library with CIELO-project Cross Sections, New Standards and Thermal Scattering Data, **Nuclear Data Sheets**, vol. 148, p. 1 – 142, 2018.

Brown, F. **Fundamentals of Monte Carlo Particle Transport**, 2005.

Chaves Barcellos, L. F., Bodmann, B. E., and Vilhena, M. T. On a comparison of a neutron Monte Carlo transport simulation to a criticality benchmark experiment, **Progress in Nuclear Energy**, vol. 134, p. 103652, 2021.

Chaves Barcellos, L. F. F. **On a continuous energy Monte Carlo simulator for neutron interactions in reactor core material considering up-scattering effects in the thermal energy region**. Master's thesis, Universidade Federal do Rio Grande do Sul, 2016.

Collaboration, G. **Book For Application Developers**, 2020a.

Collaboration, G. **Geant4 Users Guide for Toolkit Developers**, 2020b.

de Camargo, D. Q. **Um Modelo Estocástico de Simulação Neutrônica Considerando o Espectro e Propriedades Nucleares com Dependência Contínua de Energia**. PhD thesis, Universidade Federal do Rio Grande do Sul, 2011.

de Camargo, D. Q., Bodmann, B. E., de Vilhena, M. T., de Queiroz Bogado Leite, S., and Alvim, A. C. M. A stochastic model for neutrons simulation considering the spectrum and nuclear properties with continuous dependence of energy, **Progress in Nuclear Energy**, vol. 69, p. 59 – 63, 2013.

Duderstadt, J. and Martin, W. **Transport Theory**. Wiley-Interscience Publications. Books on Demand, 1979.

Feshbach, H. **Theoretical Nuclear Physics, Nuclear Reactions**. John Wiley & Sons, 1992.

Glasstone, S. and Sesonske, A. **Nuclear reactor engineering: Reactor design basics. Volume One**. Chapman and Hall, New York, NY (United States), 1994.

Guennebaud, G. **An overview of Eigen**, 2011.

Guennebaud, G. **Eigen: a c++ linear algebra library**, 2013.

Lamarsh, J. **Introduction to nuclear reactor theory**. Addison-Wesley series in nuclear engineering. Addison-Wesley Pub. Co., 1966.

Matsumoto, M. and Nishimura, T. Mersenne Twister: A 623-dimensionally Equidistributed Uniform Pseudo-random Number Generator, **ACM Trans. Model. Comput. Simul.**, vol. 8(1), p. 3–30, 1998.

Monk, S., Shippen, B., Colling, B., Cheneler, D., Hamrashdi, H. A., and Alton, T. A comparison of MCNP6-1.0 and GEANT 4-10.1 when evaluating the neutron output of a complex real world nuclear environment: The thermal neutron facility at the Tri Universities Meson facility, **Nuclear Instruments and Methods in Physics Research**

**Section B: Beam Interactions with Materials and Atoms**, vol. 399, p. 48 – 61, 2017.

Piotrowski, T., Tefelski, D. B., Polański, A., and Skubalski, J. Monte Carlo simulations for optimization of neutron shielding concrete, **Central European Journal of Engineering**, vol. 2(2), p. 296–303, 2012.

Pitts, M., Rahnema, F., Williamson, T., et al. **Water-reflected 91-liter sphere of enriched uranium oxyfluoride solution, HEU-SOL-THERM-012**. In International Handbook of Evaluated Criticality Safety Benchmark Experiments. [DVD]/NEA 7329, OECD Nuclear Energy Agency, Paris, 2018.

Reif, F. **Fundamentals of statistical and thermal physics**. Waveland Press, 2009.

Reuss, P. **Neutron Physics**. Nuclear engineering. EDP Sciences, 2008.

Sekimoto, H. Nuclear Reactor Theory, **Tokyo Institute of Technology Press, Tokyo**, vol. 11, 2007.

Sunny, E., Brown, F., Kiedrowski, B., and Martin, W. Temperature effects of resonance scattering for epithermal neutrons in MCNP, **International Conference on the Physics of Reactors 2012, PHYSOR 2012: Advances in Reactor Physics**, vol. 1, p. 803–811, 2012.

Team, X.-. M. C. **MCNP: A general Monte Carlo N-Particle transport code, version 5**, 2003.

Tolman, R. C. **The principles of statistical mechanics**. Courier Corporation, 1979.

Tran, H., Marchix, A., Letourneau, A., Darpentigny, J., Menelle, A., Ott, F., Schwindling, J., and Chauvin, N. Comparison of the thermal neutron scattering treatment in MCNP6 and GEANT4 codes, **Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment**, vol. 893, p. 84 – 94, 2018.

Williams, M. **The Slowing Down and Thermalization of Neutrons**. North-Holland, 1966.

# APPENDIX A – Additional Program Information

## A.1 Cross-Section Source Code

The program that builds the cross-section source code files operates by reading the values for the cross-section taken from ENDF/B-VIII.0 data Brown et al. [2018], and performing the linear interpolation of points. The program, however, aims to minimize the number of comparison operations required to find the function that is valid for a given neutron energy.

This can be done by recursively dividing the data in $k$ intervals, repeating this operation for $l$ recursions. Therefore, the maximum number of comparisons made to find the energy interval of the worst case scenario is given by

$$(k-1)\,l + \frac{N}{k^l} - 1,$$

in which $k$ is the division number, $l$ is the number this operation is recursively applied, and $N$ is the total number of data points. The $-1$ factors account for the fact that only the *if* statements are comparative operations, but not the final *else* clause. By using this procedure it is possible to assure that, for the case of capture in $^{238}U$, which has more than 210000 data points, a linear interpolating function can be correctly found in fewer than 30 comparisons.

## A.2 Sampling from the Maxwell-Boltzmann Distribution

In order to sample energies from the Maxwell-Boltzmann distribution effectively, the cumulative distribution function was inverted numerically. This was done by sampling the cumulative distribution function for a selected temperature in 10001 equally spaced points in the range 0 $eV$ to 1 $eV$, inverting the coordinates of these points and performing linear interpolation in all $10^4$ intervals. It is important to mention that, even though the distribution that is defined in a semi-infinite interval is being linearly interpolated in a closed interval, the value of the integral of the Maxwell-Boltzmann probability distribution function, for a typical coolant temperature of 568 $K$, between the limits 1 $eV$ and $+\infty$ is smaller than $10^{-8}$. Therefore, the integral between 0 $eV$ and 1 $eV$ can be approximated to be unit, and all meaningful information is contained in this interval.

## A.3 Random Number Generator

The chosen pseudo-random number generating algorithm was the Mersene Twister engine, originally developed by Makoto Matsumoto and Takuji Nishimura. The period is $2^{19937} - 1$ and in the compiler generation C++11 it is included as an engine of the `<random>` library. Details of the algorithm may be found in reference Matsumoto and Nishimura [1998].

## **APPENDIX B** – Data Structure

A custom data structure was created to increase the performance of saving and loading the simulated data. In this structure the saved data is divided according to the neutron interaction and the interval of Monte Carlo steps. This is done by creating a container object for the data of a neutron interaction in a given interval. This container is a member of either the *InputMatrix* or *OutputMatrix* template classes. Note that these classes do not hold data in matrices of fixed dimensions, but in vectors of vectors, where the internal vectors can have different sizes. Also *OutputMatrix* objects, as the name implies, have exclusively output operations and are used for the simulation phase of the program, whereas *InputMatrix* objects have exclusively input operations and are used for data assessment.

During a simulation *OutputMatrix* objects store the values of the *Neutron* objects for every reaction and for every monte Carlo step in the interval. At the end of the interval these values are saved as binary files, for future use. This operation consists of first reducing the size of the *DataMatrix* member, this is the container that holds the data for and *OutputMatrix* object, and then writing the output. Since container classes, like *std::vector*, cannot be directly reinterpreted as binary, the class uses the following *BinaryOutput* function to output a header that defines the construction of the *DataMatrix*, and then all neutron data serialized.

```
void BinaryOutput(const std::string filename){
  std::ofstream File;

  Reduce(); \\Operation to reduce DataMatrix size.

  unsigned int i;
  unsigned int number_of_lines;
  std::vector <unsigned int> columns_per_line( DataMatrix.size() );

  number_of_lines = DataMatrix.size();
  for( i = 0; i < DataMatrix.size(); i++ ) {
    columns_per_line[i] = DataMatrix[i].size();
  }
```

```
  File.open(filename, std::ios::binary);
  if ( !File.good() ) {
    std::cout << "Warning: BAD FILE - "
              << filename << std::endl;
  }

  File.write( reinterpret_cast <const char*> ( &number_of_lines )
            , sizeof( unsigned int )
            );
  File.write( reinterpret_cast <const char*> ( &columns_per_line[0] )
            , columns_per_line.size() * sizeof(unsigned int)
            );
  for( i = 0; i < DataMatrix.size(); i++ ){
    File.write( reinterpret_cast <const char*> ( &DataMatrix[i][0] )
              , DataMatrix[i].size() * sizeof(data_type)
              );
  }

  File.close();
}
```

When data needs to be input in the program, mainly during data assessment, InputMatrix objects are used. They reconstruct the *DataMatrix* member that was saved by the *OutputMatrix* class. This is done by reinterpreting the binary data with the following member function *BinaryInput*. Note that this function uses the information in the file header to reconstruct the *DataMatrix* container, and then read the serialized data.

```
void BinaryInput(const std::string filename){
  unsigned int i;
  unsigned int number_of_lines;
  std::vector <unsigned int> columns_per_line( DataMatrix.size() );

  File.open(filename, std::ios::binary);
  if( !File.good() ) {
    std::cout << "Warning: BAD FILE - "
```

```
                << filename << std::endl;
    }

    File.read( reinterpret_cast <char*> ( &number_of_lines )
             , sizeof( unsigned int )
             );

    columns_per_line.resize( number_of_lines );
    DataMatrix.resize( number_of_lines );

    File.read( reinterpret_cast <char*> ( &columns_per_line[0] )
             , columns_per_line.size() * sizeof( unsigned int )
             );

    for( i = 0; i < DataMatrix.size(); i++ ) {
      DataMatrix[i].resize( columns_per_line[i] );
    }

    for( i = 0; i < DataMatrix.size(); i++ ) {
      File.read( reinterpret_cast <char*> ( &DataMatrix[i][0] )
               , DataMatrix[i].size() * sizeof( data_type )
               );
    }

    File.close();
}
```

# APPENDIX C – Thread Pool

Two template classes were created, to provide a thread pool functionality. This relies in reserving a fixed number of threads to perform a certain number of jobs, and each time a thread finishes a job, it picks a new one, without having to synchronize with other working threads. Synchronization to the original calling thread happens when all computation is finished.

These classes allow for the computation of a function a given number of times, with a given number of arguments. They are initialized with a vector of *f_args* objects, a pointer to a function that will be repeatedly computed (each time with one of the *f_args* objects), and the number of reserved threads for this operation.

Two different pool classes were created, to consider the particularities of functions that have a return value, and the functions that are of the *void* type. Their main differences is the fact that the *return_pool* class posses a getter method, to retrieve a vector of *f_return* objects, and that this return vector is attuned to the arguments vector, so that each result object can be traced to an arguments object.

The complete source code of both classes is presented in this chapter. Note that they both use the following template function.

```
template <typename R> bool is_ready (std::future<R> const& f) {
  return f.wait_for( std::chrono::seconds(0)
                   ) == std::future_status::ready;
}
```

## C.1 Void Pool

```
template <typename f_args> class void_pool {

  private:
    typedef void ( *f_pointer )( f_args );

    f_pointer m_func;

    std::vector <f_args> m_args;
```

```
  int m_cores;

  int m_execs;

  bool m_runFunc( f_args arg, int exec ) {
    m_func( arg );
    return true;
  }

public:

  void_pool( std::vector <f_args> arguments
           , int cores
           , f_pointer func
           ) {
    m_args  = arguments;
    m_cores = cores;
    m_execs = arguments.size();
    m_func  = func;
  }

  virtual ~void_pool() {};

  void Run() {
    int completed_execs  = 0;
    int available_cores  = m_cores;

    int running_exec = 0;

    bool wait_return;

    std::vector < std::future <bool> > futures;
```

```
        while ( completed_execs < m_execs ) {
          wait_return = true;

          if ( (running_exec < m_execs) && (available_cores > 0) ) {
            futures.push_back( std::async( &void_pool::m_runFunc
                                          , this
                                          , m_args[running_exec]
                                          , running_exec
                                          )
                             );

            running_exec++;
            available_cores--;
          }

          else{
            while ( wait_return ) {
              for ( int exec = 0; exec < futures.size(); exec++ ) {
                if ( is_ready( futures[exec] ) ) {
                  if ( futures[exec].get() ){
                      completed_execs++;
                      available_cores++;
                      futures.erase( futures.begin() + exec );
                      wait_return = false;
                  }
                }
              }
            }
          }
        }
    }
};
```

## C.2 Return Pool

```
template <typename f_return, typename f_args> class return_pool{

  private:
    typedef f_return ( *f_pointer )( f_args );

    std::vector <f_return> m_returns;

    f_pointer m_func;

    std::vector <f_args> m_args;

    int m_cores;

    int m_execs;

    bool m_runFunc( f_args args, int exec ) {
      m_returns[exec] = m_func( args );
      return true;
    }

  public:

    return_pool( std::vector <f_args> arguments
               , int cores
               , f_pointer func
               ) {
      m_args  = arguments;
      m_cores = cores;
      m_execs = arguments.size();
      m_func  = func;
      m_returns.resize( arguments.size() );
    }
```

```
~return_pool() {};

void Run() {
  int completed_execs  = 0;
  int available_cores  = m_cores;

  int running_exec = 0;

  bool wait_return;

  std::vector < std::future <bool> > futures;

  while ( completed_execs < m_execs ) {
    wait_return = true;

    if ( (running_exec < m_execs) && (available_cores > 0) ) {
      futures.push_back( std::async( &return_pool::m_runFunc
                                     , this
                                     , m_args[running_exec]
                                     , running_exec
                                     )
                         );

      running_exec++;
      available_cores--;
    }

    else{
      while ( wait_return ) {
        for ( int exec = 0; exec < futures.size(); exec++ ) {
          if ( is_ready( futures[exec] ) ) {
            if ( futures[exec].get() ) {
              completed_execs++;
```

```
                    available_cores++;
                    futures.erase( futures.begin() + exec );
                    wait_return = false;
                  }
                }
              }
            }
          }
        }

    std::vector <f_return> Returns() {
      return m_returns;
    }
};
```