National University of Singapore

Final year project

A Monte-Carlo Simulation for Neutron Transport in Spherical Reactor

Dai Kunlun

supervised by

Prof. Lim Hock & Prof. Chung Keng Yeow

April 4, 2016

Contents

Abstract

Following the existing simulation routines of Monte-Carlo N-particle transport, we developed an original computer program to perform Monte-Carlo simulation on a spherical shaped reactor kernel. Analysis focusing on scalar flux and effective eigenvalue is carried out based on the result of the program. Results are compared with 3-energy-group deterministic calculation. The most probable neutron energy in MC method is slightly lower than the fast group energy assumed in deterministic method. Later on, the effect of enrichment and reflector are studied. With decreasing enrichment of Uranium-235 isotope, critical radius of the sphere increases from $r_c = 10.8\text{cm}$ to infinitely large. With the presence of paraffin reflector layer, much smaller critical radius is required. In particular, we showed how a subcritical reactor with 55% enrichment and $R_u = 15.0\text{cm}$ is brought to supercritical condition with thickening reflector layer.

Chapter 1

Introduction

1.1 Introduction on the Project

Neutron transport refers to the study on neutron-material interaction. It first originated from the kinetic theory of gases, but prevailed through the development of nuclear physics. Increasingly challenging problems in nuclear industry requires higher computational power and more accurate approximations. Two main branches of approaches — deterministic method and stochastic method, are used in neutron transport calculations today. In particular, Monte-Carlo simulation as a stochastic method, has been extensively used in evaluating the performance of new reactor designs. Some code currently in use include MCNP in Los Alamos National Laboratory for multiple-purpose particle interaction simulation; KENO in National Energy Agency for criticality and flux simulation[15]; Serpant in VTT Technical Research Centre of Finland for spatial homogenization and fuel cycle studies[10]. Beyond neutron transport, Monte-Carlo particle transport code such as GEANT[5], has also been applied in high energy particle experiments.

In this project, we aim to reproduce a simplified Monte Carlo simulation specifically for a single spherical fuel kernel wrapped by certain type of moderator. Specifically, the simulation results will be presented in section 4 for three cases: pure U-235 core, enriched Uranium core with variable enrichment, and enriched Uranium wrapped in paraffin layer. These three cases was also studies by Tan Yan Ren using deterministic method. Hence a comparison between the two methods will be possible. The motivation of this project is to realize some basic analysis such as cricality calculation and scalar flux plotting on the aforementioned spherical geometry. We did not have access to the official MCNP code and this whole project is carried out by writing an original code that imitates the official code in terms of routines and structures. Some of the relevant code can be found in appendices. Despite the simplicity, the model has the potential to be adopted for more complicated simulation. The spherical, layered geometry can very well describe the structure of a fuel kernel in many reactor designs. For example, generation IV reactors such as high-temperature gas-cooled reactor *Allegro* in Europe, and very-high-temperature reactor HTTR[16] in Japan both adopt such structure where the spherical fuel kernels is packed into pebble bed or prismatic blocks to form the reactor core.

1.2 Neutron Transport Equation

The neutron transport equation reads

$$\begin{aligned}\frac{1}{v}\frac{\partial}{\partial t}\psi(\vec{r},\hat{\Omega},E,t) = &- \left[\hat{\Omega}\cdot\vec{\nabla} + \sigma(\vec{r},E)\right]\psi + q_{ex}(\vec{r},\hat{\Omega},E,t) \\ &+ \int dE' \int d\Omega' \sigma_s(\vec{r},E' \to E, \hat{\Omega}'\cdot\hat{\Omega})\psi(\vec{r},\hat{\Omega}',E',t) \\ &+ \chi(E)\int dE' \nu\sigma_f(\vec{r},E')\int d\Omega' \psi(\vec{r},\hat{\Omega}',E',t), \qquad (1.1)\end{aligned}$$

where the quantity $\psi(\vec{r},\hat{\Omega},E,t) = vN(\vec{r},\hat{\Omega},E,t)$, known as the angular flux of neutron , describes the distribution of neutrons inside a reactor, It equals the speed times number density of neutrons with the corresponding traveling direction and energy at the specified position and time. It can alternatively be defined as the total of the path lengths traveled per unit time by all particles in phase volume $dVd\Omega dE$.

The change in this distribution has contributions from several physical processes, shown on the right hand side of the equation. The first term gives the net loss of neutrons in a volume element positioned at $\vec{r}$ (for a specific angular direction) due to out going flux and out-scattering. The second term is an external source. The third term gives the in-scattering of neutrons from all angles that ends up traveling in the direction $\hat{\Omega}$. And the last term is the multiplication term given rise by fission. When the angular dependence of flux is not of great concern, another quantity is commonly used:

$$\phi(\vec{r},E,t)=\int\psi(\vec{r},\hat{\Omega},E,t)d\Omega.$$

ϕ known as the scalar flux, is the integration of flux on spherical angle. When considering spherical reactors such as discussed in this paper, the quantity ϕ is enough to study how the distribution of neutron varies as a function of radial distance r.

An important aspect of neutron transport calculation is to determining the criticality condition. For this purpose, usually a time independent form of transport equation is preferred:

$$\begin{aligned}\left[\hat{\Omega}\cdot\vec{\nabla}+\sigma(\vec{r},E)\right]\psi(\vec{r},\hat{\Omega},E)=&\int dE'\int d\Omega'\sigma_s(\vec{r},E'\to E,\hat{\Omega}'\cdot\hat{\Omega})\psi(\vec{r},\hat{\Omega}',E')\\&+\chi(E)\int dE'\frac{\nu}{k}\sigma_f(\vec{r},E')\phi(\vec{r},E')\end{aligned}\tag{1.2}$$

Here no external source is considered and an additional factor in fission multiplicity is introduced. This is understood as the hypothetical adjustment needed for a steady state solution to exist. Hereby the time-dependent equation becomes an eigenvalue problem.

1.3 MCNP

Monte-Carlo N-Particle Transport Code (MCNP) is a general-purpose code to simulate the transport process of neutron, photon, electron, etc, first developed by Los Alamos National Laboratory in 1957[6]. The most up-to-date MCNP6 package allows users to switch between modes for transport calculation for different types of particles. It is suitable for typical reactor geometries like close-packed fuel pebbles and fuel kernels lattices[9].

The essential components of a MCNP code include

- geometric tracking module
- tallying[*]

[*]tallying is the terminology in MCNP for collecting data entries for statistics

- collision module
- eigenvalue calculation

Figure 1.1(a) shows the general procedure of a MCNP code. The program simulate the physical process by tracking the movement and collision of each single particle. Depending on the sequence of code, there could be two types of calculations. The first type is a fixed-source calculation. One single source and its secondary particles are tracked down until distinction(due to absorption, leakage or rejection by weigh-modifying methods). In such calculation, a source is also known as a history. The second type treats a collective of neutrons as a batch, and all neutrons in the current batch is brought through exactly one random walk to form the next batch. Such is called eigenvalue calculation since it is convenient to find eigenvalue in this way. More on eigenvalue will be discussed in section 3.2.4.

In this project we have adopted the eigenvalue calculation scheme. In addition, we choose to follow analog Monte Carlo, which is characterized by[4]

- faithful simulation of particle histories
- no alternation of PDFs
- at collision, particle is killed if absorption occurs
- particles are born with weight 1.0
- weight is unchanged throughout history until particle is killed
- score 1.0 when tallying events of interest

Thus, weight modifying schemes such as Russian Roulette are not applied. This is justified when the main interest of our program lies in critical conditions, where the neutron number inside reactor remains stabilized.

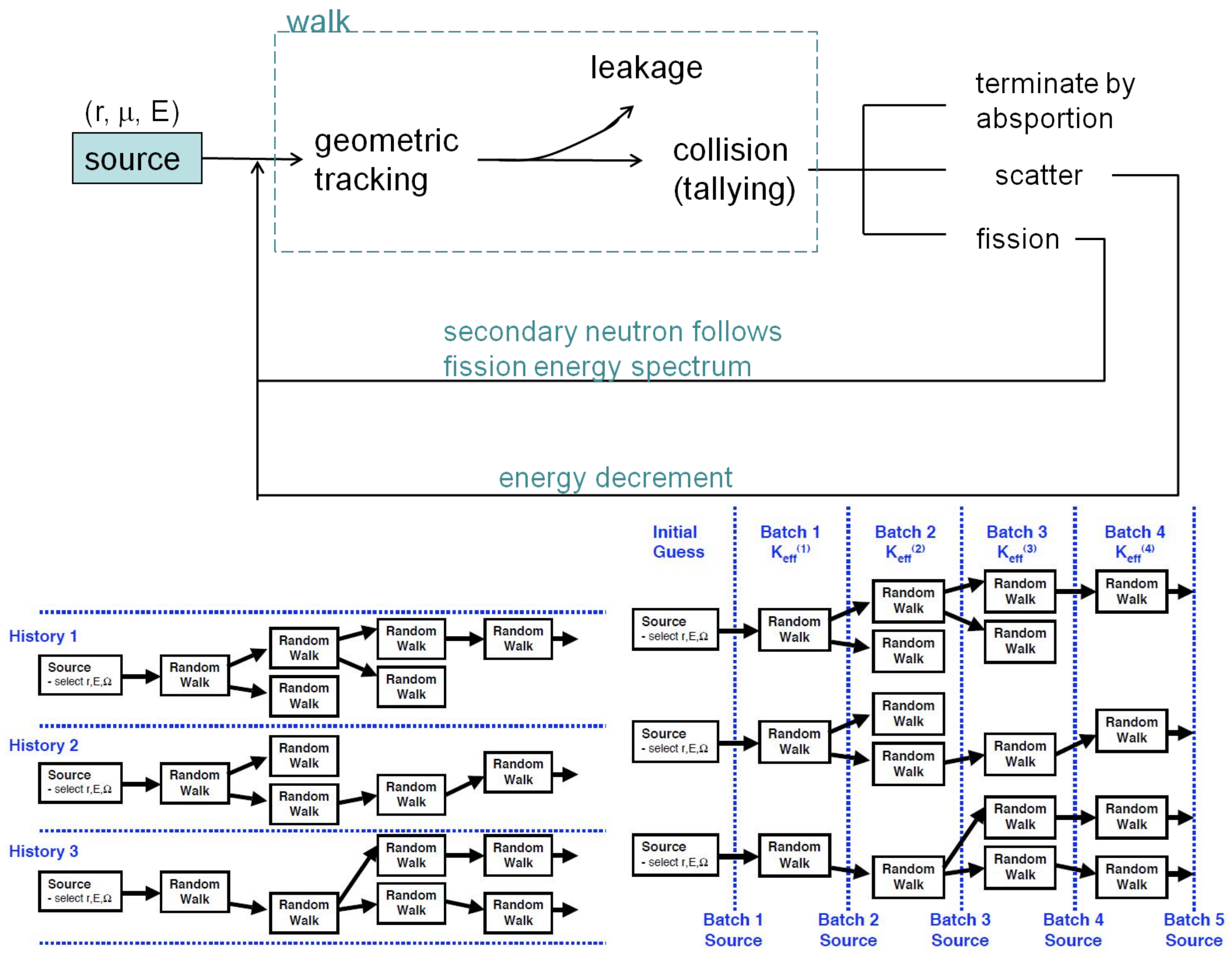


Figure 1.1: (a)Top: A flowchart for a typical MCNP code
(b)Bottom: comparison between fixed-source MC and batched MC
PICTURE FROM **Forrest B. Brown** *Fundamental of Monte Carlo Particle Transport*

Chapter 2

Preliminary Statistics

2.1 Pseudorandom Number Generator

Being able to generate statistically random numbers is essential for all type of sampling. Though genuine randomness is impossible to create using algorithms, pseudorandom number generators can produce sequences of numbers which appear to be randomly sampled from a uniform distribution, usually in the range $[0, 1)$. A most straightforward one is the '*linear congruential random number generator*'. Its sequence goes according to

$$\begin{aligned} S_{i+1} =& (gS_i + c) \mod 2^m, \\ \xi_i =& S_i/2^m, \qquad (0 < \xi_i < 1) \end{aligned}$$

Large value for g and for m are desirable to reduce serial correlation. More systematic introduction on the mathematics can be found for example in Knuth's publication[8]. The default random number generator in MATLAB uses Mersenne Twister algorithm, which is slower but can provide higher-quality randomness. The period for the most commonly used Mersenne Twister MT19937 is $2^{19937} - 1$.

In most Monte-Carlo simulations, there is a need to keep track of the seed of each step or to jump between seeds. In an eigenvalue calculation of the MCNP code, Adjacent seeds will be used for the random walks of two particles in the same batch, while a 'stride' (jump over a number of seeds) will allow one to switch between batches. For this reason, linear congruential method is much more convenient and hence chosen by the MCNP5 program.

2.2 Sampling Methods

2.2.1 Inverse sampling

Given a probability density function(PDF) of variable x,

$$P\{a \leq x \leq b\} = \int_a^b f(x)dx,$$

$$F(x) = P\{x' \leq x\} = \int_{-\infty}^x f(x')dx'.$$

Consider another variable $\xi = F(x)$, it follows that

$$f(x)dx = d\xi.$$

ξ defined in such way has a uniform probability distribution function, hence it could easily be generated by a RNG as sampling points between 0 and 1. Variable $x = F^{-1}(\xi)$ then appear as sampling points that follow the given PDF.

2.2.2 Rejection sampling

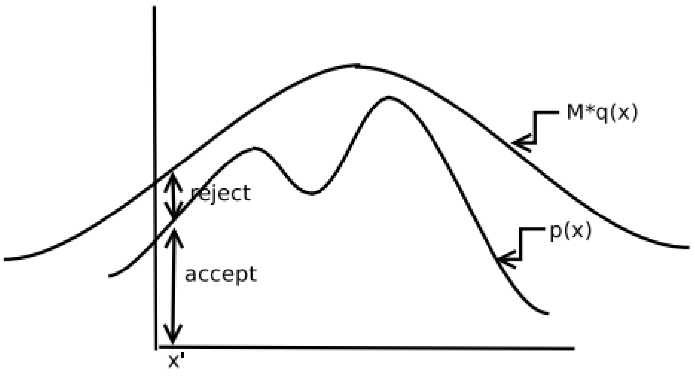


Figure 2.1: illustration on rejection sampling.
PICTURE BY **M. Jordan** AND **S. Jain**, 2010, *Lecture notes on Monte Carlo sampling*

Rejection sampling is a commonly used technique to sample a variable with relatively complicated distribution, in cases where inverse sampling may become impossible or overly difficult. Rejection sampling is itself a type of Monte-Carlo method. In figure 2.1, $p(x)$ is the actual distribution of variable x, and $Mq(x)$ is some other scaled distribution whose support includes the support of $p(x)$*, $q(x)$ should be easy to sample using inverse sampling, and M is a constant that ensures $p(x) \leq Mq(x)$ for all values of x.

The sampling procedure starts by generating a sample point x' from $q(x)$:

$$x' = Q^{-1}(\xi_1), \qquad Q(x) = \int q(x),$$

followed by a comparison between a second random number ξ_2 and the ratio $\dfrac{p(x')}{Mq(x')}$.
if $\xi_2 Mq(x') \leq p(x')$, $\rightarrow$ accept x' as a sample point;
if $\xi_2 Mq(x') > p(x')$, $\rightarrow$ reject this x', regenerate $x' = G^{-1}(\xi_1)$.
The envelope principle based on the relative size of area under probability distributions guarantees that x' sampled this way follows distribution $p(x')$.

2.3 Central Limit Theorem

In statistics, the Central Limit Theorem states that for a large sample of independent variables x_n with well-defined expectation value and variance, taking their arithmetic mean

$$\hat{x} = \frac{1}{N}\sum_{n}^{N} x_n$$

results in a quantity that is approximately normally distributed, regardless of the underlying distribution of x. Specifically, if variable x has expectation value $\bar{x}$, the expectation value of $\hat{x}$ is also $\bar{x}$

$$E[\hat{x}] = E\left[\frac{1}{N}\sum_{n=1}^{N} x_n\right] = \frac{1}{N}\sum_{n=1}^{N} E[x_n] = E[x] = \bar{x}.$$

*this statement says that $q(x) > 0$ whenever $p(x) > 0$

The variance of $\bar{x}$ follows

$$\sigma^2(\hat{x}) = E\left[\left\{\frac{1}{N}\sum_{n}^{N}(x_n - \bar{x})^2\right\}^2\right] = \frac{1}{N^2}\sum_{n=1}^{N} E[(x_n - \bar{x})^2] + \frac{1}{N^2}\sum_{n \neq n'}\sum E[(x_n - \bar{x})(x_{n'} - \bar{x})]$$

$$\sigma^2(\hat{x}) = \frac{1}{N}\sigma(x)$$

Averaging on a large number N of histories therefore gives a valid estimation on the population mean. But to obtain the population variance from finite number of samples (without knowing the exact $\bar{x}$) requires the use of unbiased sample variance.

$$S^2 = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \hat{x})^2 = \frac{N}{N-1}(\widehat{x^2} - \hat{x}^2).$$

This guarantees that

$$E[S^2] = \frac{1}{N-1}\left\{\sum_{n=1}^{N} E[(x_n - \bar{x})^2] - NE[(\hat{x} - \bar{x})^2]\right\} = \sigma^2(x).$$

Furthermore, the distribution of $\hat{x}$ becomes normal for large N.

$$f_N(\hat{x}) = \sqrt{\frac{N}{2\pi}}\frac{1}{\sigma(x)}\exp\left[\frac{-N(\hat{x} - \bar{x})^2}{2\sigma^2(x)}\right], \qquad n \to \infty.$$

Chapter 3

MCNP for spherical symmetric reactor

3.1 Cross Sections

3.1.1 Definition

Microscopic cross section σ measures the cross-sectional area of a single nuclei for the type of scattering considered. It gives the scattering rate, hence the intensity decrement per unit length in material of the incident beam.

$$\frac{d}{dx}I(x) = -N\sigma I(x). \tag{3.1}$$

Here N is the number density of nuclei inside material, usually has the unit cm^{-3}. Microscopic cross-section σ usually takes the unit of barn. 1 barn = 10^{-24} cm^{-2}.
Naturally, a quantity

$$\Sigma = N\sigma = \frac{\rho N_0}{A}\sigma$$

called macroscopic cross section is more relevant in most calculations. Clearly, in earlier equations (1.1),(1.2), σ actually means the macroscopic cross section Σ. In other sections of this paper, we will always use σ for macroscopic cross section and any use of microscopic cross section will be specially mentioned.

When considering homogeneously mixed material, the resultant macroscopic cross section will simply be the sum of that due to each type of substance $\Sigma = \sum_i N_i\sigma_i$. For mixtures whose component proportion is specified by mole fraction, such as enriched Uranium, the following equation is useful

$$\Sigma = \frac{\rho N_0}{A_{\text{eff}}} \sum_i e_i \sigma_i \tag{3.2}$$

Here $e_i = \dfrac{N_i}{N}$ is the enrichment of the ith element/isotope, and $A_{\text{eff}} = \sum_i e_i A_i$ is the effective mass number of the mixture.

For mixture whose component proportion is specified by mass fraction, such as mixture of different substances, the following equation is useful

$$\Sigma = \rho_{\text{eff}} N_0 \sum_i \frac{\alpha_i \sigma_i}{A_i} \tag{3.3}$$

Here $\alpha_i = \dfrac{M_i}{M} = \dfrac{\rho_i V_i}{\rho V}$ is the mass fraction of the ith substance and $\rho_{\text{eff}} = (\sum_i \dfrac{\alpha_i}{\rho_i})^{-1}$ is the effective density of the mixture.

3.1.2 Reaction types

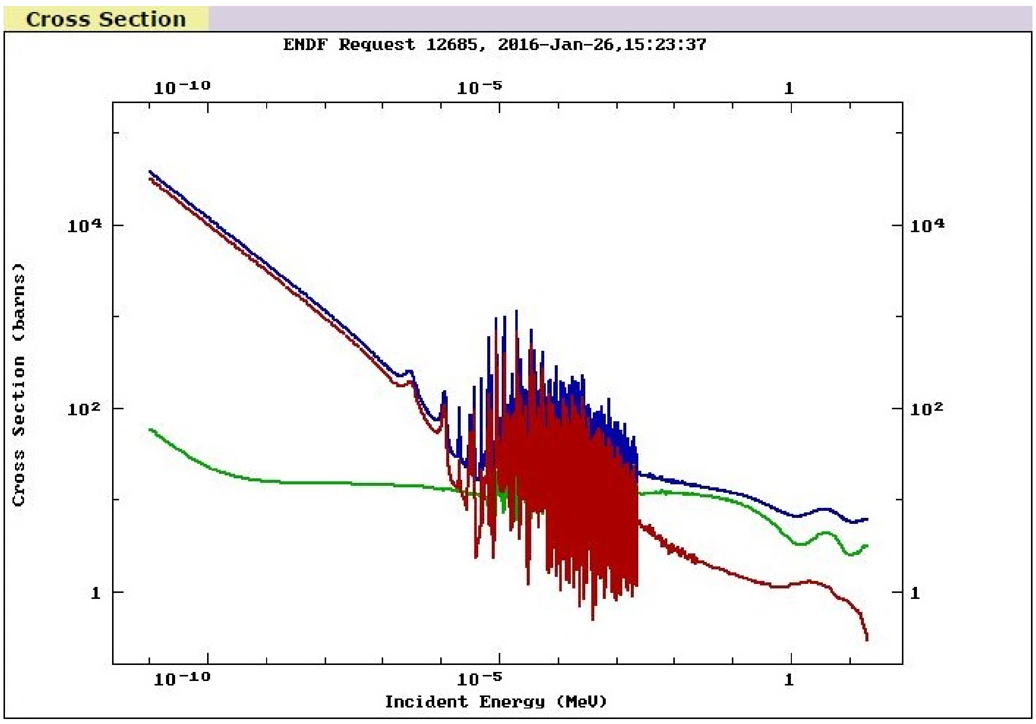
A collision between neutron and nuclei may lead to different types of reactions, namely, scatter, capture and fission. To reflect the relative probability of happening of each of these processes, the total cross section is divided into scattering and absorption cross sections $\sigma = \sigma_s + \sigma_a$. And absorption is divided further into capture and fission cross sections $\sigma_a = \sigma_\gamma + \sigma_f$. Thus given a collision, the probability that the neutron is scattered, captured or triggers a fission are respectively $\dfrac{\sigma_s}{\sigma}, \dfrac{\sigma_\gamma}{\sigma}, \dfrac{\sigma_f}{\sigma}$.

3.1.3 Obtaining cross section data

The cross sections are dependent on the energy of the incoming neutron. The Evaluated Nuclear Data File(ENDF)[1] provides an ideal source for the data of microscopic cross sections at different energy. Figure 3.1(a) below is a plot of the microscopic cross section of Uranium-235 against energy obtained from ENDF. Total, scattering and fission cross sections are respectively drawn in blue, green and red.

The original data is unnecessarily numerous for the purpose of this project, and consists of a segment of rapidly changing region of resonance in the intermediate energy range. These resonance are due to matching neutron kinetic energy with quantum states of the nuclei. To incorporate the energy dependence conveniently into our program, an interpolation using cubic spline method with 21 interpolation points in thermal range and 10 interpolation points in fast range is applied. The interpolation result is drawn next to the original plot as figure 3.1(b).

Similar interpolations have been done to U-238 and other elements. The graphs can be found in appendix A.



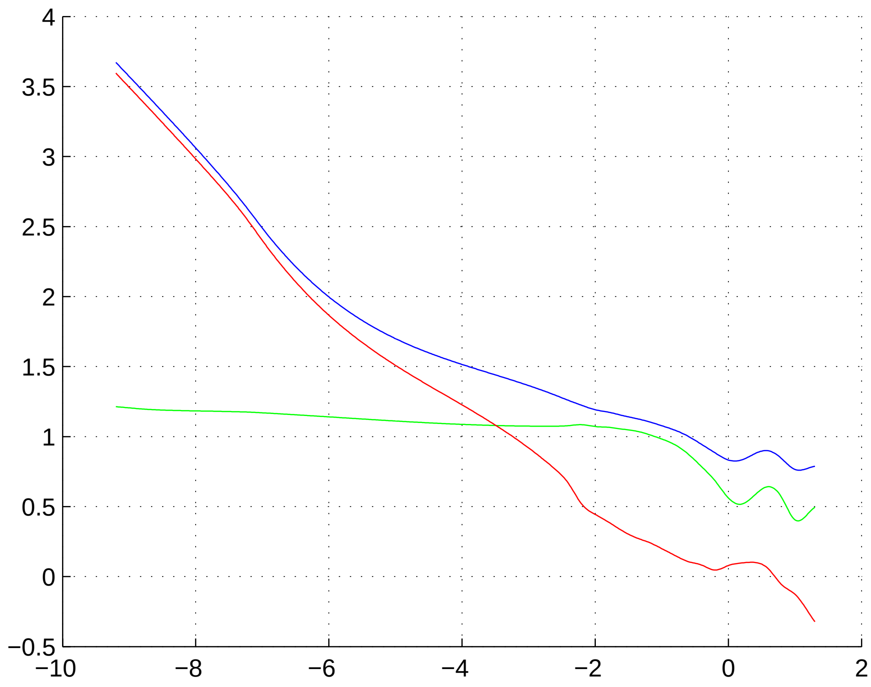


Figure 3.1: (a)log-log graph of cross sections of U-235 in the energy range 10^{-10} to 10 MeV. Original data provided by ***Young, Chadwick, Talou, Madland, Leal*** in ENDF/B-VII.1 in 2011
(b)interpolation of cross sections of U-235 from 10^{-9} to 10 MeV using cubic spline

3.2 Geometric Tracking

3.2.1 Coordinates

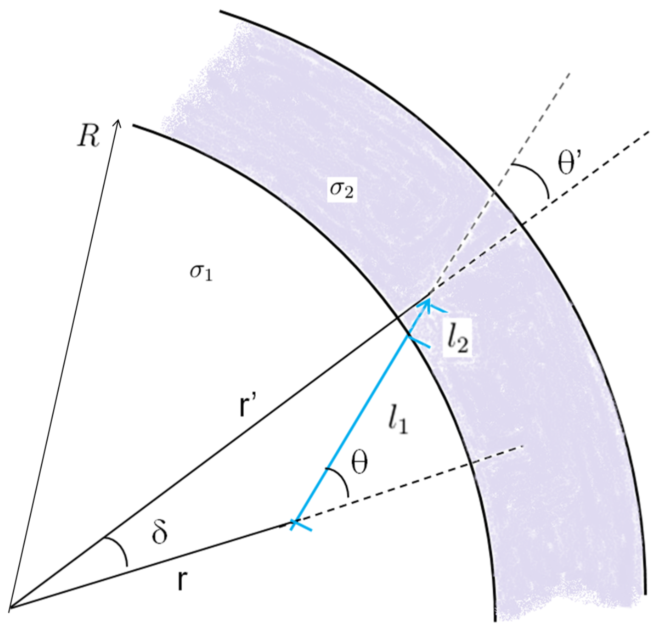


Figure 3.2: one random walk that goes across the layer boundary. The parameters (r, θ) are updated to be (r', θ') after the walk.

In a reactor with spherical symmetry, flux $\psi(\vec{r}, \hat{\Omega}, E)$ has spatial dependence that is only relevant to distance from centre of sphere, and angular dependence that is only relevant to the angle between traveling direction $\hat{\Omega}$ and radial direction $\hat{r}$. That is, ψ and its directional gradients can be rewritten as

$$\begin{aligned}\psi(\vec{r}, \hat{\Omega}) &= \psi(r = |\vec{r}|, \mu = \hat{\Omega} \cdot \hat{\mathbf{r}}) \\ \hat{\Omega} \cdot \nabla\psi(r, \mu) &= \mu\frac{\partial\psi(r, \mu)}{\partial r} + \frac{1-\mu^2}{r}\frac{\partial\psi(r, \mu)}{\partial\mu}\end{aligned} \tag{3.4}$$

In the spherical reactor Monte-Carlo code, three parameters r, μ, E are registered for each neutron. Geometric tracking is responsible to update the first two parameters whenever a neutron performs a random walk. The geometry can be illustrated by figure(3.2). The blue line represents the path of this walk. A neutron started in inner layer with cross-section σ_1 has initial parameters (r,$\mu = \cos\theta$,E). At the end point of the blue arrow, the neutron undergoes a collision event. Its spatial parameters there becomes r' and $\mu' = \cos\theta' = \hat{\Omega} \cdot \hat{\mathbf{r}}'$. While how its energy change will depend on the type of collision happened.

It can be easily checked that the final (r', θ') are related with the initial (r, θ) by

$$\begin{aligned} r' &= (r^2 + l^2 + 2rl\cos\theta)^{1/2} \\ \cos\delta &= \frac{1}{2rr'}(r^2 + r'^2 - l^2) = \frac{1}{r'}(r + l\cos\theta) \\ \theta' &= \theta - \delta \end{aligned} \tag{3.5}$$

where l, the path length of this random walk, will be discussed immediately.

3.2.2 Mean free path

Essential in simulating the neutron movement correctly is the estimation of mean free path between two subsequent collisions. Following equation (3.1), the mean free path of a neutron inside material with cross-section σ follows an exponential decay

$$F(l) = 1 - e^{-\sigma l}$$

By inverse sampling,

$$l = -\frac{1}{\sigma} ln(1 - \xi)$$

follows the correct distribution of path length x, where ξ is random number generated between 0 and 1.

In the case where Uranium is wrapped by a paraffin layer, The path may go across the boundary between these two materials, and the previous result based on constant cross-section will not be correct. Instead, a quantity called optical path length

$$L = \sum_i \sigma_i l_i = -ln(1 - \xi)$$

is used, and is understood as the addition of physical path lengths inside different materials weighted by the respective cross-sections. Refer to figure(3.2) again, where a typical boundary-crossing random walk is drawn. The following procedure is used to find l:

- generate optical path length $L = -ln(1 - \xi)$.

- calculate the maximal physical length l_1 the neutron could travel in its current medium σ_1.

$$l_1 = -\cos\theta \pm \sqrt{R^2 - r^2 \sin^2\theta}$$

(+) if $r < R$, neutron is going outward from the inner sphere;

(−) if $r > R$, neutron is going inward from the outer sphere.

- if $L < \sigma_1 l_1$, neutron collides before it reaches the boundary, $l = L/\sigma_1$.
- if $L > \sigma_1 l_1$, neutron crosses the boundary, $l = l_1 + \dfrac{L - l_1\sigma_1}{\sigma_2}$.

In the last step, extra complication may be involved if the neutron goes inward from outer shell, piercing through a path in inner shell and reenter the outer shell again. The detailed treatment can be found in the actual code of 'function MCtest2E_ walk' in appendix C.

3.2.3 Tallying

Tallying is the terminology used in MCNP for accumulation of data. Properties such as collision density and path length can be used to calculate the scalar flux.

Collision estimator

Inside a given volume $\tilde{V}$, a simple relation exists between the collision density and the average scalar flux $\bar{\phi}$. The mean number of collision is simply given by the total cross section times scalar flux $\bar{c} = \tilde{V}\tilde{\sigma}\bar{\phi}$. Using Monte Carlo method to estimate the mean collision number normalized to 1 source neutron $\hat{c} = \frac{1}{N}\sum_n c_n$, $\hat{c}$ approaches $\bar{c}$ for large N. Therefore

$$\hat{\phi} = \frac{1}{\tilde{V}\tilde{\sigma}}\frac{1}{N}\sum_n c_n. \tag{3.6}$$

- $\tilde{V}$: volume element
- $\tilde{\sigma}$: average cross section in volume $\tilde{V}$
- c_n: number of collisions made in $\tilde{V}$
- $\hat{\phi}$: estimated average scalar flux in $\tilde{V}$

Path length estimator

One disadvantage of collision estimator is that a neutron may in fact contribute to the flux in $\tilde{V}$ even though it collides outside the volume, vice versa. The path length estimator, on the other hand, tallies the path segments every particle contributes when passing through $\tilde{V}$. Since scalar flux can be defined as total track length traversed by all particles per unit volume per unit time(section 1.2), we have the path length estimator

$$\bar{\phi} = \frac{1}{\tilde{V}} \frac{1}{N} \sum_n l_n. \tag{3.7}$$

Quantities bear the same meaning as in collision estimator, with l_n being the track length in $\tilde{V}$ made by the n^{th} particle.

3.2.4 Eigenvalue

In introduction section 1.2, the neutron transport equation had been cast into the form of an eigenvalue equation. A additional factor $\frac{1}{k}$ is used to scale the fission multiplicity. This approach is useful in deterministic calculation as it reduces the transport equation into a linear system

$$\begin{aligned}
\mathbf{\Psi_{j+1}} &= M\mathbf{\Psi_j}, \\
M &= (\hat{\Omega} \cdot \vec{\nabla} + \sigma)^{-1}(\mathcal{F} + \mathcal{S})
\end{aligned}$$

$\mathcal{F}$ and $\mathcal{S}$ simply refer to the fission and scattering integrals. Hereby, (the largest) eigenvalues of M determines the criticality of the reaction.

Deterministic method starts from a trial solution

$$\mathbf{\Psi_0} = \sum_k C_k \psi_k.$$

Assuming the largest eigenvalue of matrix M, $\lambda_1 = \max\{\lambda_i\}$ is non-degenerate. As long as $\mathbf{\Psi_0}$ has non-zero component in ψ_1, after a large number of iterations, the flux will eventually approaches the eigenstate $\mathbf{\Psi_j} \approx \lambda_1^j C_1 \psi_1$.

With this understanding, we can conveniently define the eigenvalue in a similar fashion for MCNP

$$k = \lim_{j\to\infty} k_j = \lim_{j\to\infty} \frac{\Phi_{j+1}}{\Phi_j}, \quad \Phi_j = \frac{1}{V}\sum_i V_i \int dE\phi_j(r_i, E).$$

j represents the iteration index and i the radial index. $\phi_j(r_i, E)$ is the (average) value of scalar flux at radius r_i. The eigenvalue k defined this way is consistent with equation(1.2).

As a matter of fact, it is hard for one to accurately obtain the eigenvalue in MCNP, since the asymptotic value undergoes constant fluctuation due to "the statistical noise inherent in the random walks" (Brown etc., 2007). Nonetheless, relevant researches have developed methods to test for convergence using Shannon entropy[3]. This technique is, however, beyond the scope of this project and not adopted in our program.

In an alternative point of view that takes into account of neutron kinetics, the ratio between subsequent batch of neutron must reflect the relative value between the fission and absorption probabilities, with some correction introduced from leakage:

$$k = P_{\text{NL}} k_\infty = P_{\text{NL}} \frac{\nu\sigma_f}{\sigma_a},$$

P_{NL} is called the non-leakage probability, and ν again is the multiplicity of fission. The kinetics is seen by bringing back the time parameter.

$$\begin{aligned} P_{\text{NL}}\frac{d}{dt}n(t) =& P_{\text{NL}}S(t) + P_{\text{NL}}(\nu\sigma_f)\bar{v}n(t) - \sigma_a\bar{v}n(t) \\ \frac{d}{dt}n(t) =& S(t) + \frac{k-1}{l}n(t) = \frac{\rho}{\Lambda}n(t). \end{aligned} \tag{3.8}$$

where $\bar{v}$ is the average velocity of neutron population considered, $S(t)$ a resource term, $l = P_{NL}/\bar{v}\sigma_a$ has a unit of time and represents the neutron lifetime[11], $\rho = \dfrac{k-1}{k}$ is the reactivity, and $\Lambda = \dfrac{l}{k}$ means the prompt generation time, which is the average time from a prompt neutron emission to a capture that results only in fission. This equation known as neutron balance equation merely restates the transport equation without looking at the spatial details.

Eigenvalue k indicates the criticality condition of the reactor:

$$k \begin{cases} > 1 & \text{supercritical} \\ = 1 & \text{critical} \\ < 1 & \text{subcritical,} \end{cases}$$

but gives little information about reaction rate. The latter is related with the fraction of delayed neutron in the population. Further discussion on delayed neutron will continue in section fission 3.3.2.

3.3 Collision Physics

The possibility of honest simulating the energy change of the neutron population characterizes an advantage of MCNP over deterministic method. The latter finds numerical solutions of the flux by discretizing energy into groups, hence making the cross-section energy dependence inaccurate. Also, the inter-group down scattering caused by loss of neutron kinetic energy may turn out to be troublesome in deterministic calculation. The following sections show the standard technique in MCNP involving the neutron energy change through different reactions.

As mentioned in section 3.1.2, different reactions happen with different probability after collision, depending on the cross-sections at where the collision happens and at the incident neutron energy. Reaction type can hence be determined by generating a random number $0 < \xi < 1$:

$$\begin{aligned} \xi < \frac{\sigma_s}{\sigma} \quad &\rightarrow \text{ neutron is scattered} \\ \frac{\sigma_s}{\sigma} < \xi < \frac{\sigma_s + \sigma_f}{\sigma} \quad &\rightarrow \text{ neutron induces a fission} \\ \xi > \frac{\sigma_s + \sigma_f}{\sigma} \quad &\rightarrow \text{ neutron is absorbed} \end{aligned}$$

3.3.1 Scattering

Scattering event changes the travelling direction of neutron as well as decreases its kinetic energy. In the centre of mass frame, we idealize the scattering to be a hard-sphere scattering. To find the angular dependence of the cross section, consider an electron colliding into an atom

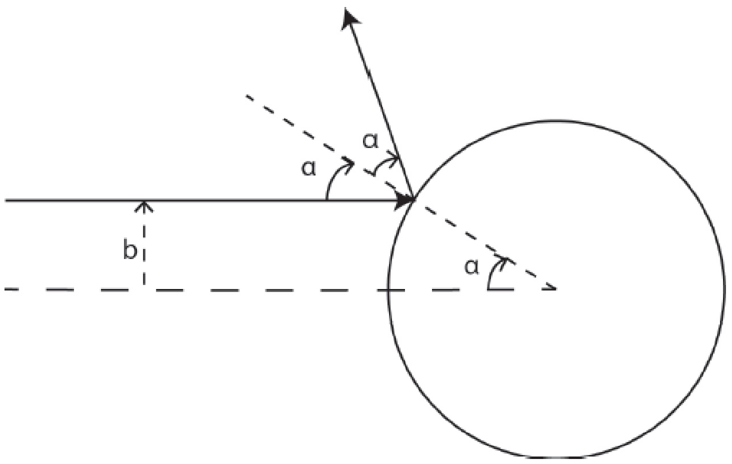


Figure 3.3: hard sphere scattering
GRAPH BY **Tan Yan Ren**

depicted as the sphere in figure 3.3. The scattering angle $\theta = \pi - 2\alpha$. Its cosine

$$\chi = \cos(\pi - 2\alpha) = 2\sin^2\alpha - 1 = 2(\frac{b}{R})^2 - 1$$

is also the dot product between the two velocity unit vectors before and after the scattering. Due to the azimuthal symmetry, we introduce the angular dependence as $\sigma_s(\chi) = \sigma_s f(\chi)$. The notation is a bit confusing due to limited symbols. But we can easily identify $\sigma_s(\chi) = \sigma_s(\hat{\Omega} \cdot \hat{\Omega}')$ as the cross section appearing in neutron transport equation 1.1. And σ_s is simply its average value over angle cosine χ. Thus $f(\chi)$ is the probability distribution function of χ. By matching cross section differentials, we have the following relation

$$\frac{2\pi b}{\pi R^2} db = f(\chi) d\chi, \qquad f(\chi) \equiv \frac{1}{2}$$

After transforming back to the lab frame, the scattering angle would appear narrowed.

$$\begin{aligned}\tan\theta_{lab} &= \frac{\sin\theta_{CM}}{\cos\theta_{CM} + m/M} = \frac{\sin\theta_{CM}}{\cos\theta_{CM} + 1/A} \\ \chi_{lab} &= \frac{1 + A\chi_{CM}}{\sqrt{A^2 + 2A\chi_{CM} + 1}}\end{aligned}$$

This provides a convenient method to generate the scattering angle in lab frame. From here on the change in neutron traveling direction upon scattering can be simulated with the correct probability.

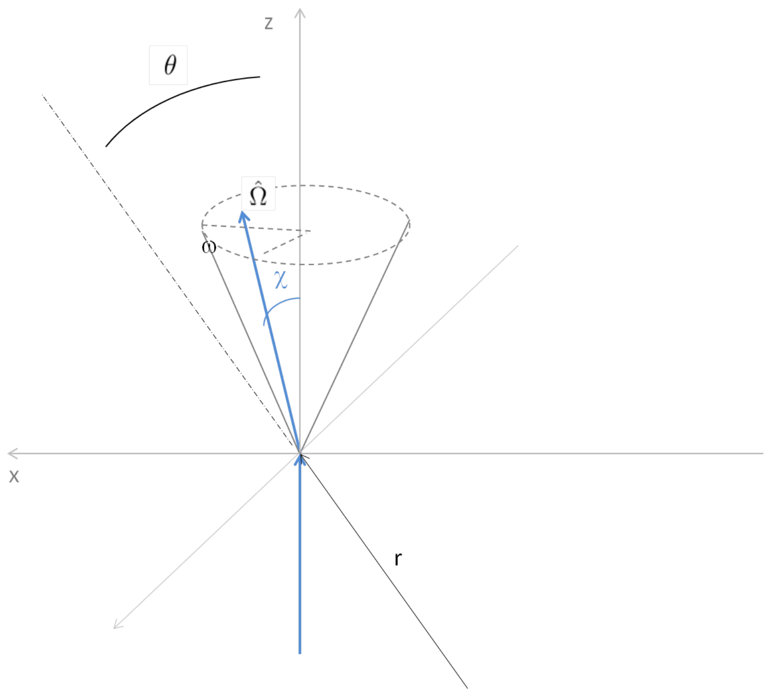


Figure 3.4: angle change upon scattering

In Figure 3.4, the two blue lines shows two segment of paths before and after a scattering event. The traveling direction $\hat{\Omega}'$ before scattering is taken as the z direction, and the line connecting scattering point and center of sphere lies in the x-z plane. The scattering angle is renamed χ to differentiate it from the angle between $\hat{z}$ and $\hat{r}$. Another azimuthal angle ω is temporarily used to find new direction $\hat{\Omega}$. Again due to symmetry, ω is uniformly sampled in $[0, 2\pi]$.

$$\begin{aligned} \hat{r} &= \cos\theta\ \hat{z} + \sin\theta\ \hat{x} \\ \hat{\Omega} &= \cos\chi\ \hat{z} + \sin\chi\cos\omega\ \hat{x} + \sin\chi\sin\omega\ \hat{y} \\ \cos\theta' &= \hat{r}\cdot\hat{\Omega} = \cos\theta\cos\chi + \sin\theta\sin\chi\cos\omega \end{aligned} \tag{3.9}$$

then gives the new $\cos\theta'$ we need as a coordinate parameter after the scattering.

For more general geometry tracking without such high symmetry, three parameters are required to represent the direction of neutron and the corresponding calculation is[12]

$$\begin{aligned}
\Omega'_x &= \frac{\sin\chi}{\sqrt{1-\Omega_z^2}}[\Omega_y \sin\omega - \Omega_y\Omega_x \cos\omega] + \Omega_x \cos\chi, \\
\Omega'_y &= \frac{\sin\chi}{\sqrt{1-\Omega_z^2}}[\Omega_x \sin\omega - \Omega_z\Omega_y \cos\omega] + \Omega_y \cos\chi, \\
\Omega'_z &= \sin\chi\sqrt{1-\Omega_z^2}\cos\omega + \Omega_z \cos\chi.
\end{aligned}$$

Finally, the energy change for the neutron is considered for elastic scattering. Using conservation laws, one can arrive at the following formula

$$E' = \frac{E(A^2 + 2A\chi_{cm} + 1)}{(A+1)^2} \tag{3.10}$$

This implies that incidental neutron with energy E might end up with energy in range $\alpha E \leq E' \leq E$ with constant probability

$$p(E \to E') = \frac{1}{(1-\alpha)E},$$

where $\alpha = (\frac{A-1}{A+1})^2$. A quantity called 'slowing down decrement' can then be naturally defined as

$$\xi \equiv \overline{\ln(E/E')} = \int_{\alpha E}^{E} \ln(E/E')\frac{1}{(1-\alpha)E}dE' = 1 + \frac{\alpha}{1-\alpha}\ln\alpha. \tag{3.11}$$

Slowing down decrement is independent of the initial and final energy and only dependent on atomic mass of the nuclear. In $A >> 1$ limit, $\xi \approx \frac{2}{A+2/3}$.

3.3.2 Fission

Energy released in a fission of heavy nucleus is mainly divided between the kinetic energy of the fission products and the fission neutrons. The energy distribution of the fast neutrons released in a fission event are commonly represented either by a Maxwellian distribution or a Watt Distribution. The latter is considered more accurate than the former by taking into account the kinetic energy of the fission fragments.[13]

A semi-empirical formula for the Watt spectrum of U-235 is[18]

$$P(E) = 0.4865 \sinh(\sqrt{2E}) e^{-E} \ \ \mathrm{MeV}^{-1}.$$

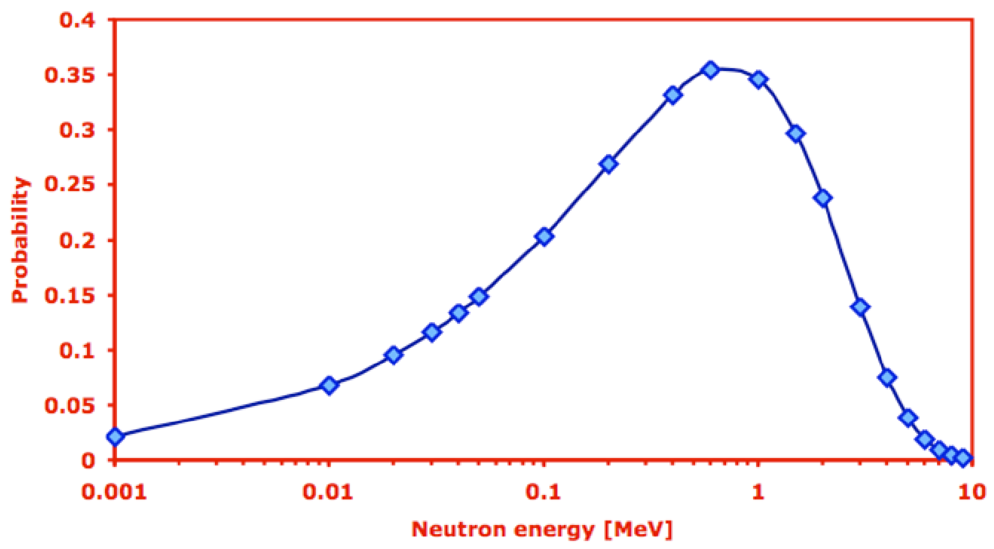
This formula is formally consistent with a slightly more general one[7].

$$P(E) = \frac{2e^{-ab/4}}{\sqrt{\pi a^3 b}} e^{-E/a} \sinh \sqrt{bE}$$

In the latter equation, two additional parameters a and b are related to the mean neutron energy and average kinetic energy of the fission fragments[4]. The former equation is more concise and hence is used in our code. Figure 3.5 below illustrate the shape of energy spectrum obtained through a rejection sampling with 100000 population size.

Another detail to be dealt with is the fission multiplication, which works against absorption and leakage to maintain the neutron population. The average fission multiplication is around 2.43 for for U-235 and 2.70 for U-238[1]. The numbers remain nearly constant throughout thermal and intermediate range of the energy spectrum, and only starts to rise significantly when the incoming neutron carries an energy of the order 10 MeV (figure 3.6)[2]. Fission products also dictate that there are either 2 or 3 neutrons release for each fission. Hence the distribution of multiplication number is taken as discrete. Namely, in 53.6% cases of all fissions, 3 neutrons are released; In other 46.4%, 2 neutrons are released. When the enrichment of U-235 is varied, multiplication number is taken as that of U-235 for convenience. This might be a source of error in the simulation, but it is considered a valid approximation since the cross-section of U235 is much larger than that of U238 for the energy ranged concerned. In general, more than 99% of the secondary neutrons will be prompt neutron, while the remaining are delayed neutrons coming from precursors with distinct liftime. Their presence modifies the neutron balance equation (3.8) to

$$\begin{aligned} \frac{d}{dt} n(t) &= S(t) + \frac{\rho - \beta}{\Lambda} n(t) + \sum_i \lambda_i C_i(t) \\ \frac{d}{dt} C_i(t) &= \frac{\beta_i}{\Lambda} n(t) - \lambda_i C_i(t), \qquad i = 1, 2, ..., 6. \end{aligned} \tag{3.12}$$



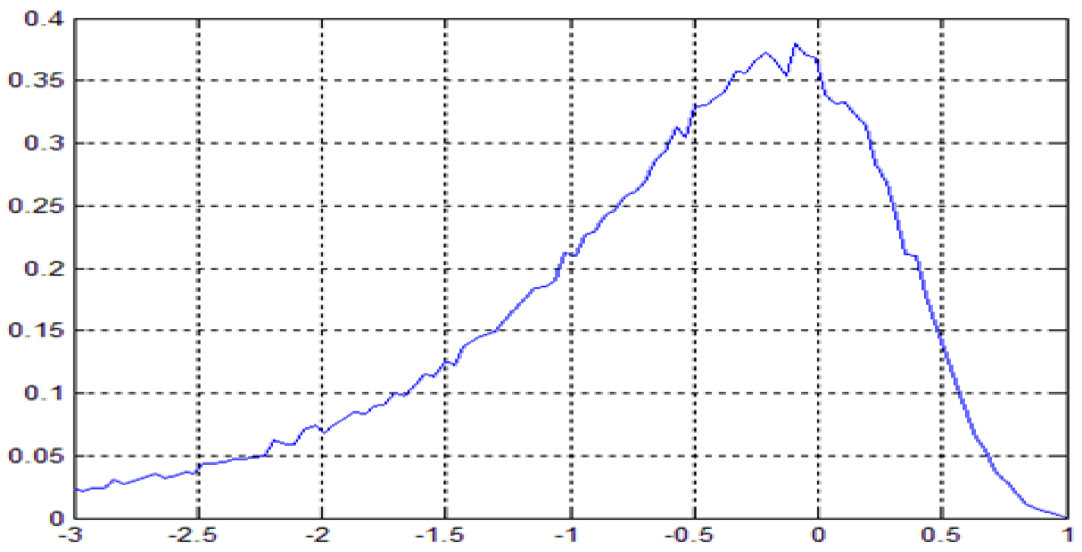


Figure 3.5: (a)Fission spectrum following Watt distribution.
PICTURE TAKEN FROM **J. Watterson**, 2007, *lecture 31 on Fission Spectrum*
(b)Fission spectrum generated through rejection sampling using 100000 population size

Here C_i represents the number of precursors of type i, whose decay rate is λ_i and whose delayed neutron fraction is β_i. Obviously delayed neutron has no effect on steady state solution, but they can significantly affect the multiplication rate of the neutron population, and thus the power generated inside the reactor.

The solution to this neutron balance equation with delayed neutron assumes a form

$$n(t) = \sum_{j=7}^{7} A_j \exp(\omega_j t).$$

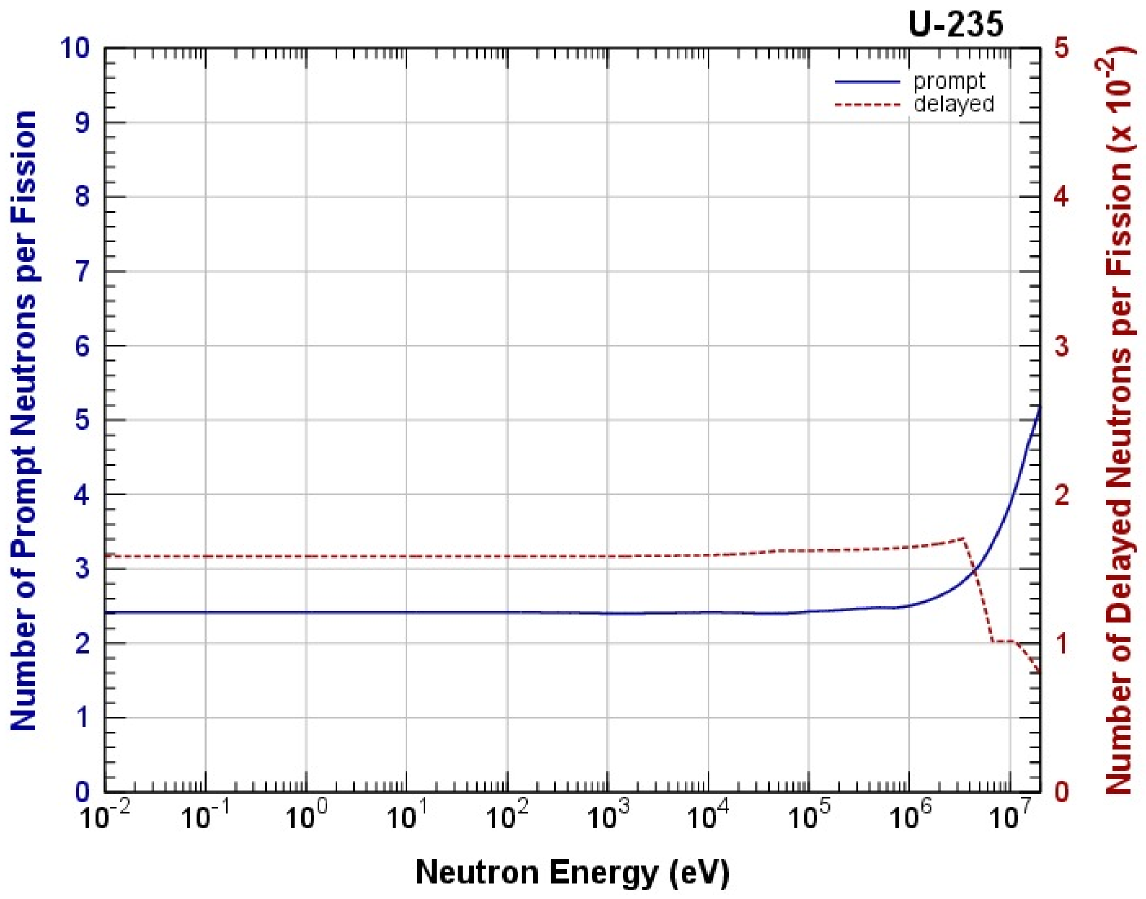


Figure 3.6: graph showing average secondary neutron number for prompt and delayed neutron separately as a function of primary neutron energy.
Figure retrieved from **JAEA Nuclear Data Center**, *JENDL-4.0 data file*

The exponential powers are can be solved numerically using the famous 'in-hour equation'

$$\rho = \left(\Lambda + \sum_i \frac{\beta_i}{\omega + \lambda_i} \right) \omega. \tag{3.13}$$

Our program skips on the extra calculation relating to delayed neutron, for they do not have effect on steady state solution. Therefore each secondary neutron is represented by adding an extra entry in the neutron list for next batch. These fission neutrons share the same spatial position r as the parent neutron that induces fission, but their energies are individually generated according to the Watt distribution and their traveling direction θ are generated isotropically. There are, however, several standard way of calculating delayed neutron fractions and powers in MCNP[14, 19]. To include such calculations appear to be a suitable future extension for this project.

Wener (2002) from *Los Alamos National laboratory* reported an accurate modeling of delayed neutrons using Spriggs method[17] and has shown a power calculation using the following formula.

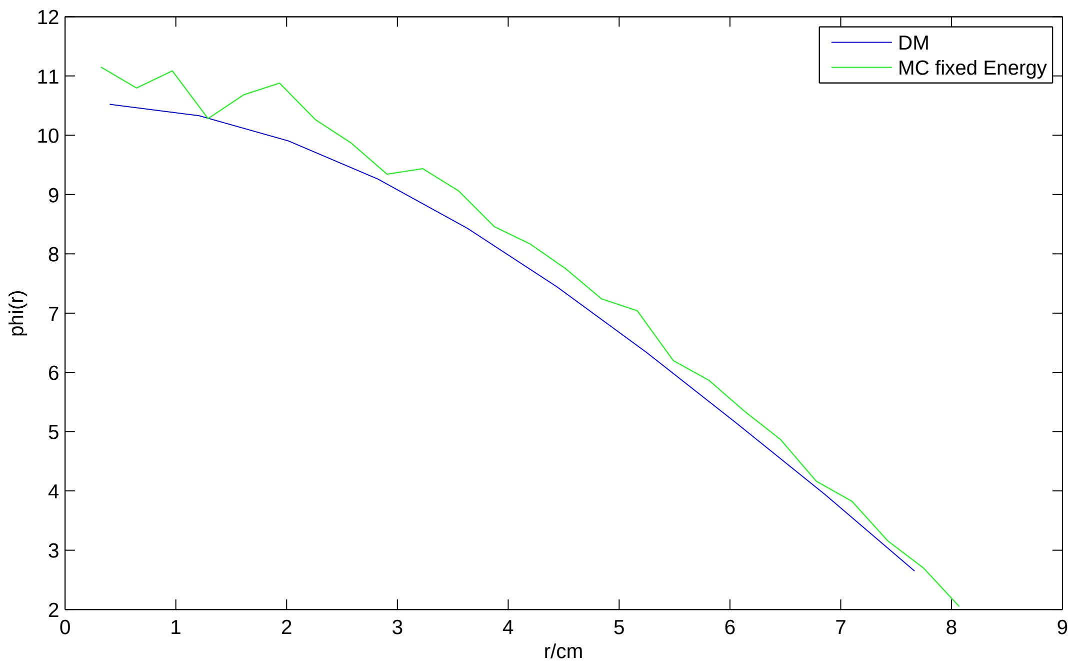
$$i(t) = l \sum_{j=1}^{m+1} \frac{e^{\omega_j t}}{l + \sum_i \frac{\beta_i \lambda_i}{(\omega_j + \lambda_j)^2}}$$

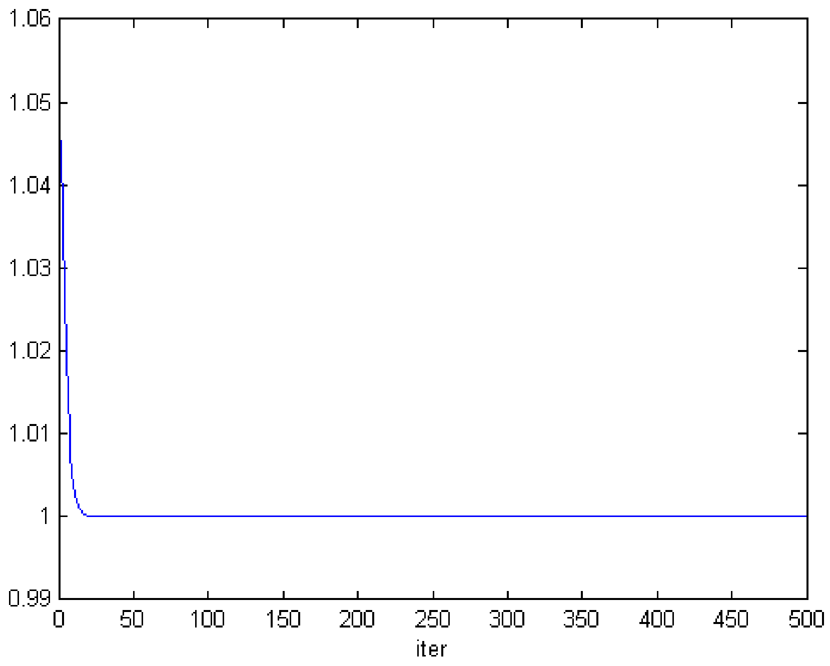
Chapter 4

Simulation Results

4.1 Pure U-235 sphere

It can be seen that among the three parameters (r, μ, E) describing the kinetics of neutrons, the two spatial parameters have been carefully revised in almost every step of simulation. While the energy profile of the batch seldom change drastically. In fact, a simulation without energy concern can also be valid when only fast neutrons are involved. This will be assuming that all neutron do not undergo enough times of scattering before it triggers a fission or escapes or gets absorbed. As a first example, a reactor made of pure U-235 sphere is considered. The simulation result using constant cross-section is compared to the steady solution obtained from deterministic method by Tan Yanren, whose method assumes constant microscopic cross-sections $\sigma = 7.62$ barn, $\sigma_s = 6.33$ barn, $\sigma_f = 1.22$ barn in his program.


DM
MC fixed Energy
phi(r)
r/cm
0
1
2
3
4
5
6
7
8
9
10
11
12


iter
0.99
1
1.01
1.02
1.03
1.04
1.05
1.06
0
50
100
150
200
250
300
350
400
450
500

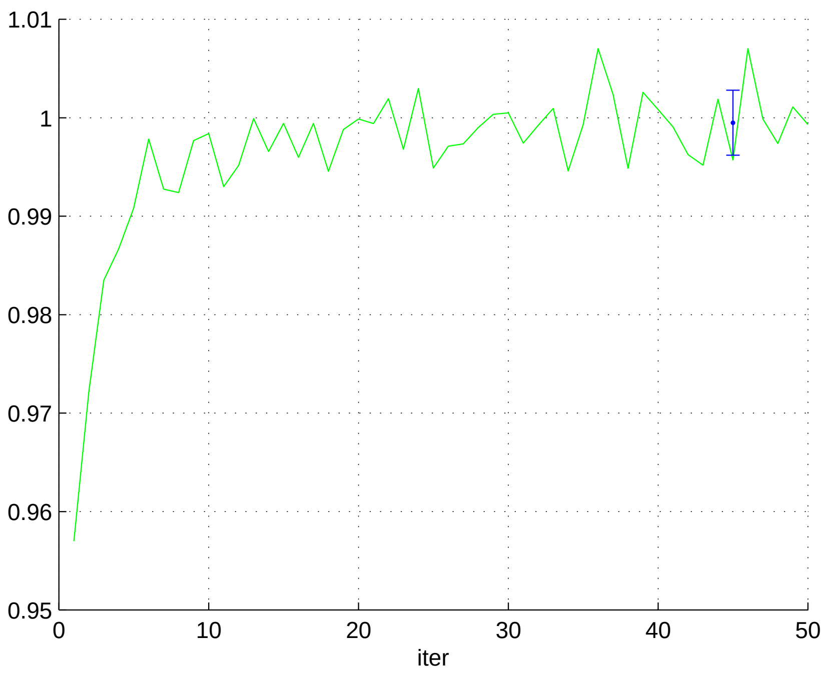


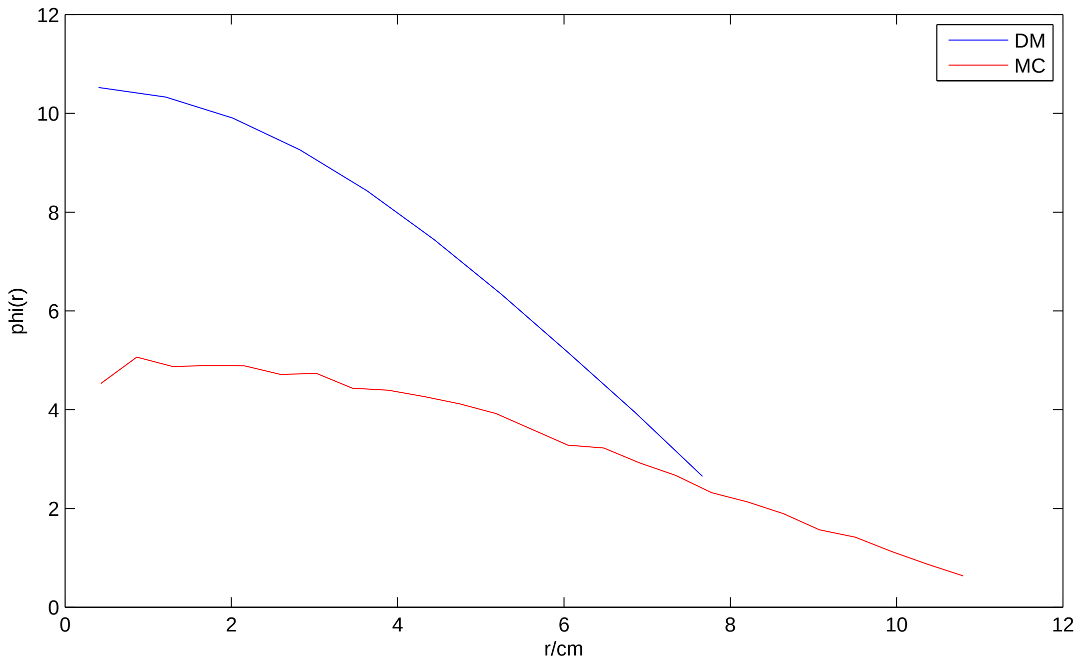
Figure 4.1: comparison between scalar fluxes (top) and eigenvalues (bottom two) obtained from deterministic calculation by Yanren (2016) and from MCNP in this project. Monte-Carlo is based on 100000 starting neutrons

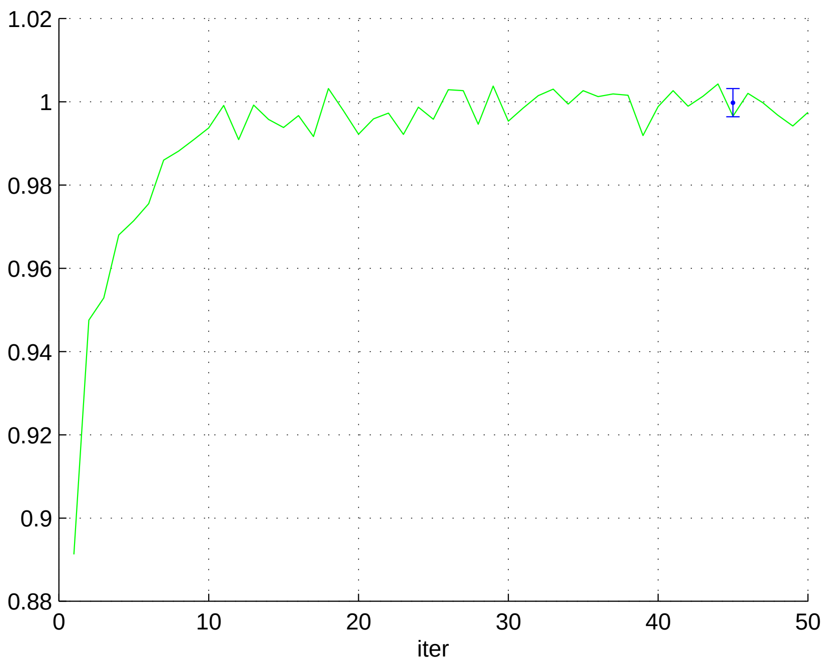
Figure(4.1) shows a direct comparison between two methods of studying neutron distribution inside the U-235 spherical reactor. The critical radius of the spherical reactor is determined to be $r_c = 8.07$cm from deterministic method. The exact same radius is then used in MC method to check for consistency.

Monte-Carlo method distribute 100000 neutrons uniformly inside the sphere in the first batch. After 50 iterations, the last batch contains 64405 neutrons. The scalar flux calculated based on the collision estimator of the last batch is drawn in green in figure 4.1(a). Deterministic method starts with a constant function $\phi(r) = 1$ as the trial solution. After 500 iterations, the final solution converges as the blue curve. The green curve of MC flux has been carefully scaled to match the blue curve. This scaling is necessary due to the lack of normalization in both fluxes. The scaling factor is calculated by requiring the total flux inside sphere to be equal. The flux from MC method in green appear to be slightly larger everywhere, this discrepancy might be the result of the error in shell volumes.

In both methods, the eigenvalue is calculated by taking the ratio between total scalar fluxes from two subsequent iterations $k_j = \lim_{j\to\infty} \dfrac{\Phi_{j+1}}{\Phi_j}$. It can be seen that the deterministic method converges very fast. Its eigenvalue has reached 1 after roughly 25 iterations. For this reason, we only performed 50 iterations in MC method to save calculation time. The eigenvalue plot for MC (green) suggests that convergence has been achieved despite of the small iteration number. It also suggests an asymptotic value of $k \approx 1$. This confirms the consistency between the two methods. Using the k_j data for $26 \leq j \leq 50$, the mean and standard error for eigenvalue is calculated to be $k = 0.9995 \pm 0.0033$.

From this step on, the energy dependence of cross-sections can be introduced. As such, using the energy modifying techniques mentioned in section 3.3.1, the MC method can simulate the energy change of neutron more accurately than the energy group approximation of deterministic method. Simulation is run on the same physical problem and the radius of the U-235 sphere is changed to recover critical condition. The new critical radius is $r_c \approx 10.8$cm. The scalar flux based on the last batch of neutron (red curve) is drawn together with deterministic solution (blue curve) in figure 4.2(a). The initial batch again contains 100000 neutrons and the ending batch has 71383 left. The mean and standard error for the eigenvalues between 25th and 50th iteration is $k = 0.9998 \pm 0.0034$.





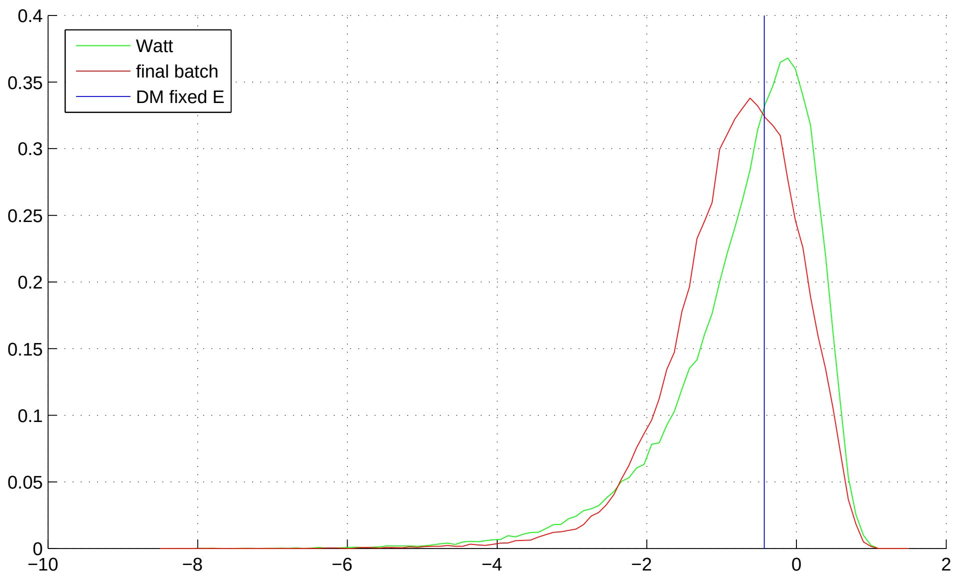


Figure 4.2: Monte-Carlo simulation result for r=11.0cm reactor, showing the scalar flux (a), eigenvalue (b) and spectrum (c) under critical condition

The energy spectrum is shown in figure 4.2(c). The green curve is the Watt distribution that fission neutrons would follow. The red curve shows the actual energy distribution for the last batch. There is a clear decrease of peak energy due to scattering energy decrements. The red curve also seem to be less skew than the original distribution. Almost all neutrons

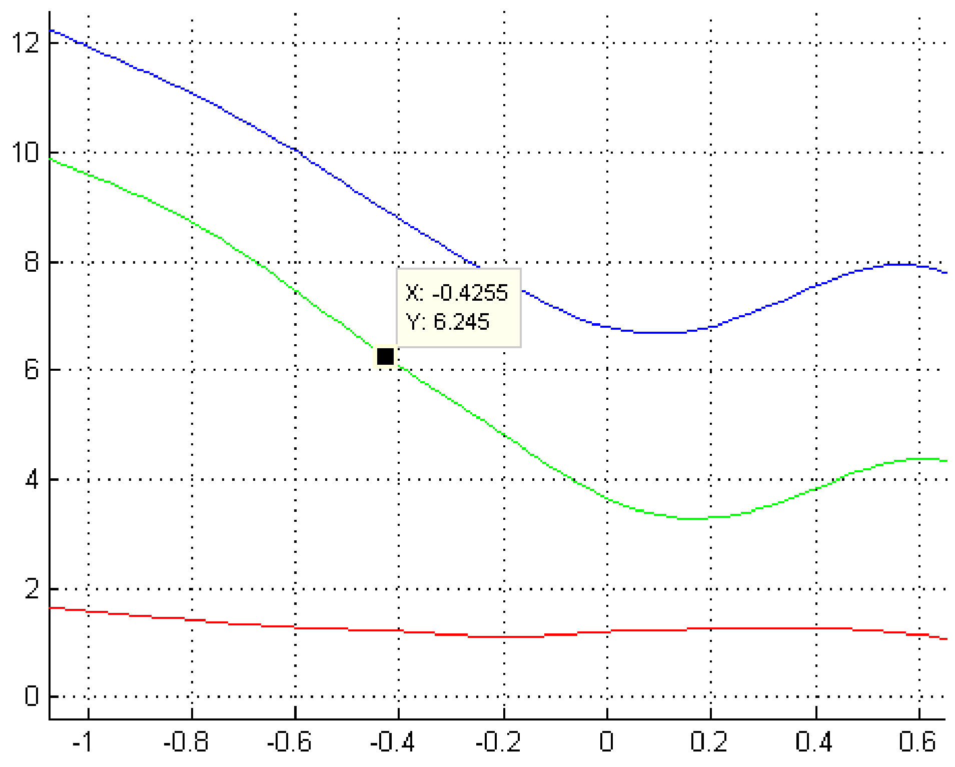


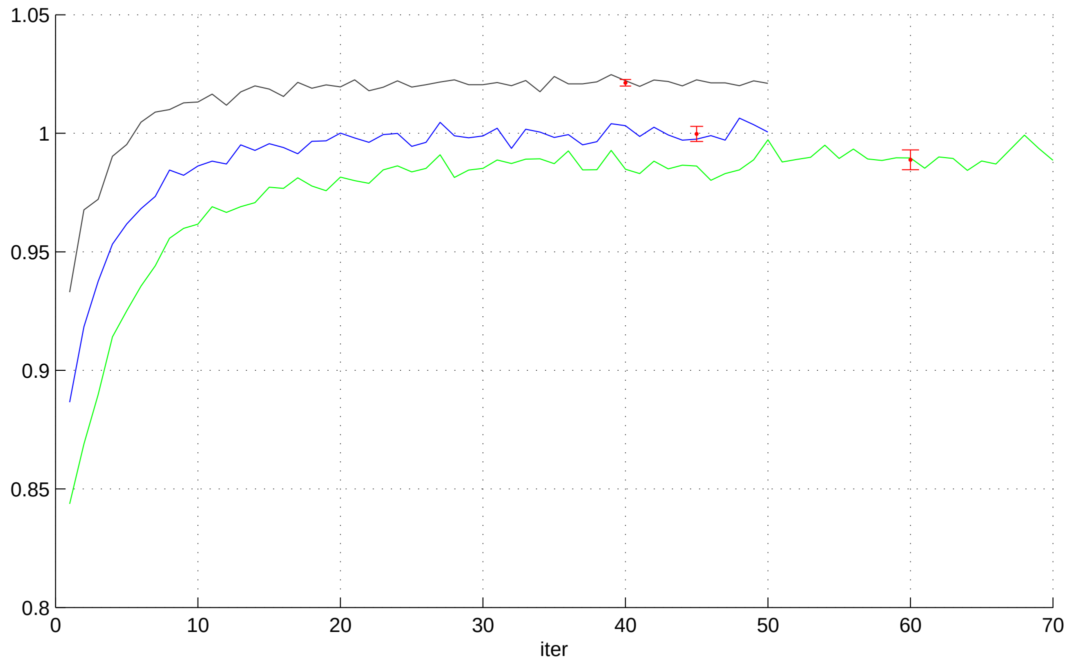
Figure 4.3: interpolation graph shows that when cross-section has value $\sigma_s = 6.33$barn, the energy is at value $\lg(E/\text{MeV}) = -0.43$

are within the fast energy range, hence the assumption used in deterministic calculation is confirmed. Furthermore, with the help of our cross-section interpolation graph, we can check the energy value corresponding to the constant cross-sections used in the previous simulation ($\sigma_s = 6.33\text{barn}, \sigma_f = 1.22\text{barn}$). This energy is around $10^{-0.43} = 0.37$MeV and is marked out as a blue vertical line. The most probable energy in last batch is slightly lower. This explains why the critical radius has increased after we introduce the energy dependence — in the $0.1 \sim 1$ MeV range, an energy decrease will cause the scattering cross-section to increase at a faster rate than the fission cross-section. As a result, multiplication effect of the system is suppressed. The leakage l goes up and eigenvalue k goes down in equation(3.8). a larger radius is needed to raise the non-leakage probability to recover the critical condition.

4.2 Enriched Uranium sphere

A direct modification on the previous problem will be to substitute the pure U-235 sphere by an enriched Uranium sphere. Cross-sections are still assumed uniform throughout the sphere. Its value is calculated according to equation(3.2). Several interpolations on natural Uranium cross-sections at different enrichment can be found in appendix A.

The program provides a straightforward way to check how the eigenvalue changes with enrichment of Uranium. We have chosen a fixed radius $r = 50.0$cm for reactor while varying the enrichment between three values $e = 30\%, 50\%, 70\%$. Eigenvalue over 50 to 70 iterations and the energy spectrum of the last neutron batch are analyzed. Scalar fluxes are not drawn as the fission is generally not critical, the amount of flux will continue to build up or fade down after more and more iterations, causing their value to differ by magnitudes.



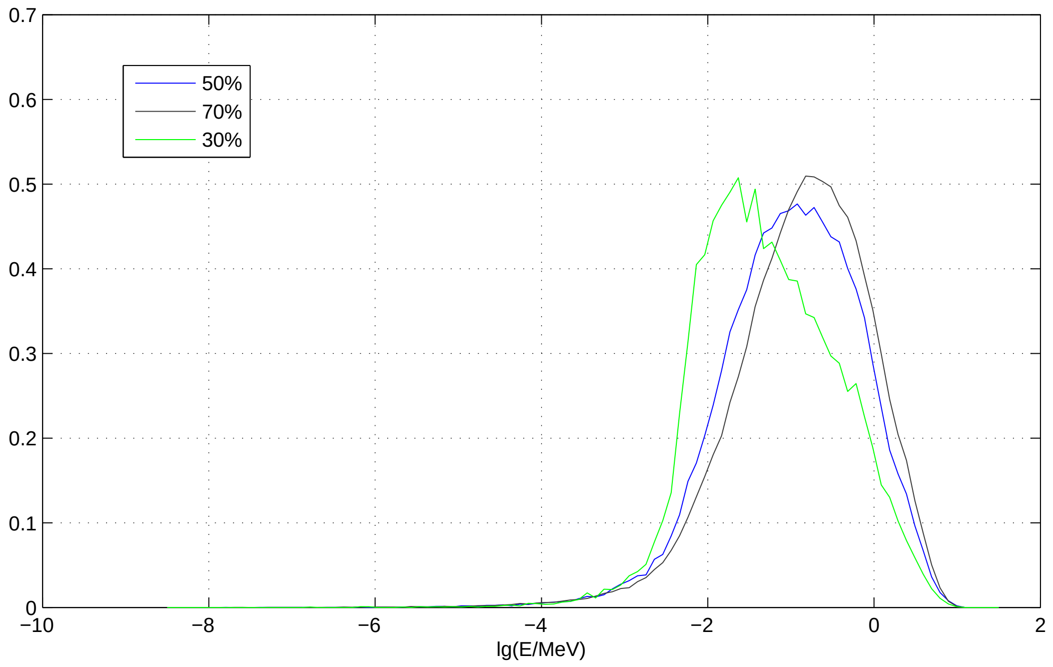


Figure 4.4: different enrichment at fixed reactor radius $R = 50.0\text{cm}$. brown: $e = 0.7$, blue: $e = 0.5$, green: $e = 0.3$

A quantitative comparison is given by studying the asymptotic behavior of the eigenvalues. The eigenvalue converges to $k = \{1.0213 \pm 0.0014,\ 0.9997 \pm 0.0032,\ 0.9888 \pm 0.0042\}$ respectively for enrichment $e = \{0.7,\ 0.5,\ 0.3\}$.

The energy spectrum of the last batch of neutron reflects how many scattering a neutron undergoes before absorption or leakage. The most probable energy value clearly decreases as the enrichment goes down. This is intuitive as U-235 has higher cross-section than U-238 and accounts for most of the fission events.

Another result obtained from this model is the relation between critical radius and enrichment. For several enrichment values from $e = 1.0$ to $e = 0.5$, a rough value of critical radius is found by adjusting its value until eigenvalue converges to 1. Table 4.1 below summarizes the critical radius and asymptotic eigenvalue at their corresponding enrichment. The eigenvalue plots based on which we determine criticality are all shown in appendix B.

enrichment e	critical radius R_c/cm	eigenvalue k
1.00	10.8	1.0024±0.0035
0.90	12.4	0.9994±0.0029
0.80	14.9	1.0001±0.0034
0.70	19.0	1.0007±0.0028
0.60	25.2	1.0002±0.0023
0.55	32.0	0.9999±0.0026
0.50	50.0	1.0006±0.0031

Table 4.1: table showing critical radius and estimated eigenvalue for reaction to be critical at various enrichments

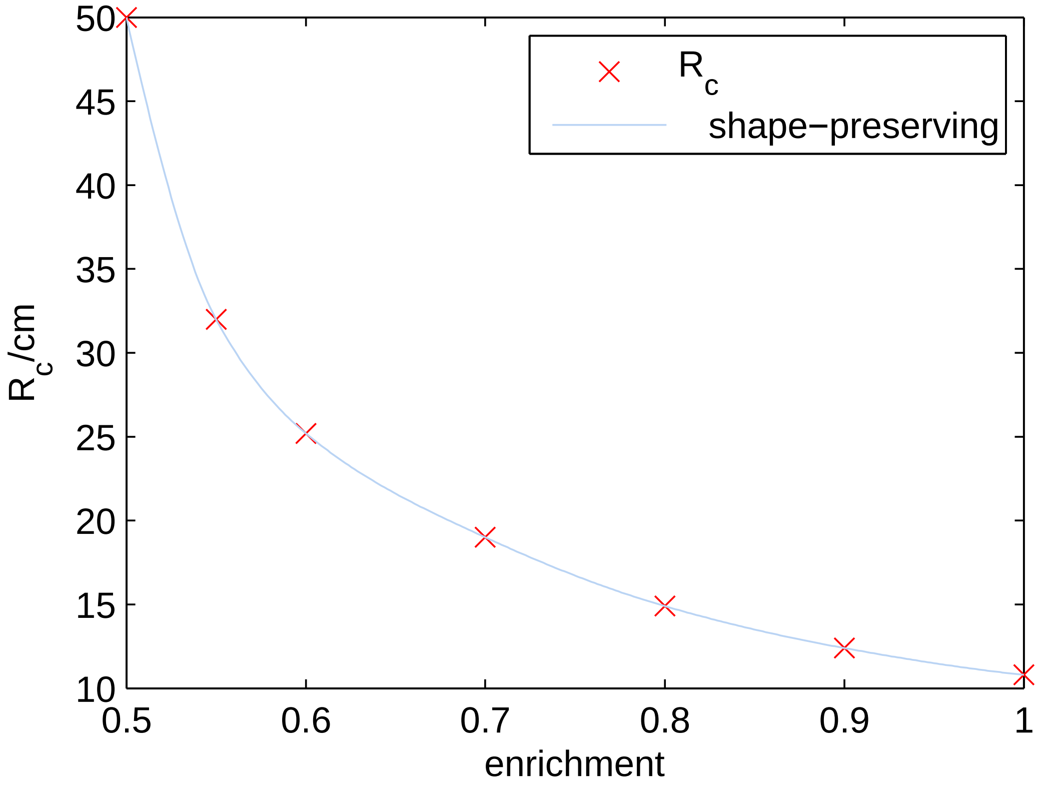


Figure 4.5: critical radius R_c against enrichment

In figure 4.5, predicted critical radius is drawn against enrichment to illustrate the trend. For $e < 0.5$, our method is unable to find an accurate enough critical radius, either because it would be too large or because the system simply never goes critical. The result, however, seems to seriously overestimate the critical radius. For highly enriched U-235 sphere, the critical mass is known to be 49.12 ± 0.15 kg (H. C. Paxton, 1975). This converts to a critical radius of 8.555 ± 0.008 cm. which is smaller than our critical radius at $e = 1.0$. The reason for this discrepancy may be due to our crude simulation on the average neutron multiplication number.

As mentioned in section 3.3.2, the multiplication ν shows a significant rise in energy range 1 10 MeV. Judging from the energy spectrum in figure 4.4, a non-negligible proportion of neutron fall in this range. Thus, the multiplication power of the reactor must have been underestimated and the critical radius obtained would be unnecessarily large. Also, the semi-empirical formula for fission spectrum might not be accurate enough. Fissions with two secondary-neutron and three secondary-neutron should give different fission spectra, but the difference is not correctly distinguished here. Nonetheless, the result does show qualitatively that a limit on enrichment level exists, below which the chain reaction can not be self-maintaining.

4.3 Uranium in paraffin reflector layer

In reactor designs, the core is usually either embedded in moderator or covered by a layer of reflector, which consists of non-multiplying material with high scattering cross-sections. Reflectors serve to scatter escaping neutrons back into fissile core material, hence significantly reduce the neutron loss due to leakage. This allows reactor to achieve critical condition at lower multiplication power, i.e. smaller critical mass. An example problem considered here is to use paraffin (C_nH_{2n+2}) reflector shell to wrap the enriched Uranium core. The effect of this paraffin layer is studied by varying its thickness.

We consider particularly the core with enrichment $e = 55\%$ and Uranium core radius $R_u = 15.0$cm. With information from the previous part, this radius at this enrichment level is subcritical. We then wrap the Uranium with a layer of paraffin of width w, so that the radius of the total sphere becomes $R = R_u + w$. Table 4.2 below shows at various paraffin shell width, what will the average scalar flux inside Uranium core (after 60 iterations) and eigenvalues be. Figure 4.6 draws three of these results on the same graph.

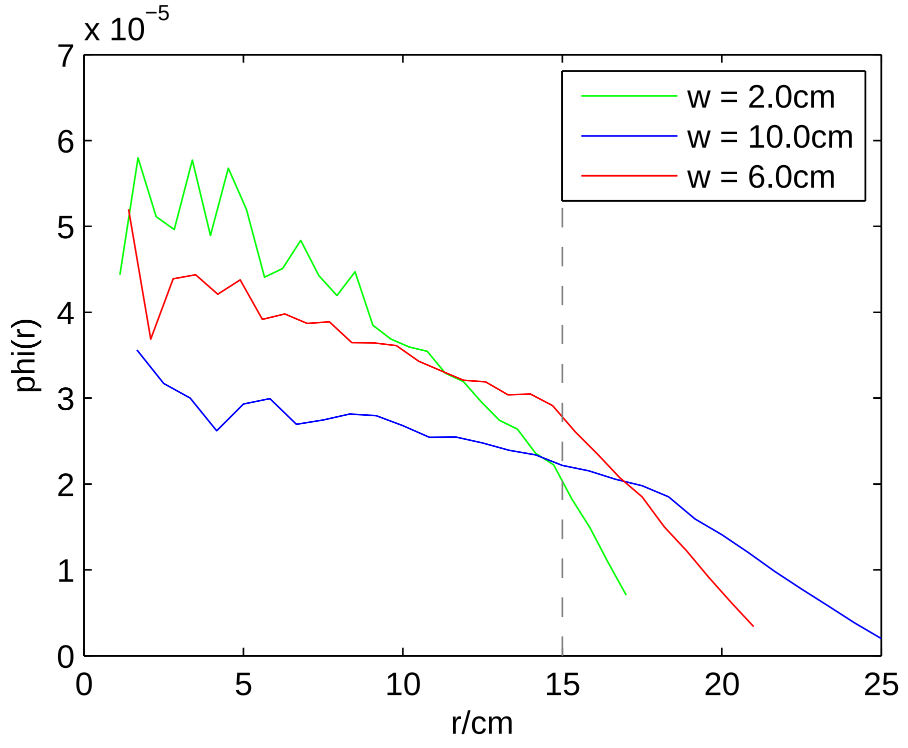
In figure 4.6(a), the fluxes clearly appear to be flattened out by the presence of reflector. The green curve corresponding to subcritical reaction at $w = 2.0cm$ has more neutrons concentrated in the central region, but the flux drops drastically neat the boundary of Uranium core. At critical condition represented by red curve, the back-scattering from paraffin layer effectively help maintain the concentration of neutron inside the core. The red flux is seen to exceed green

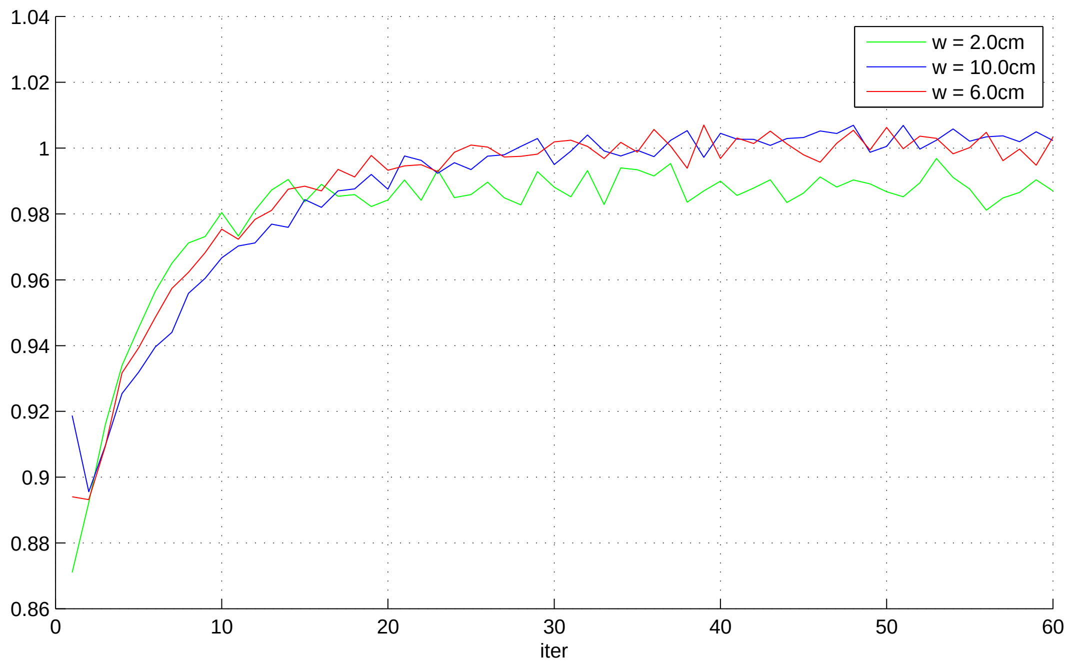
R/cm	w/cm	$\phi_{60}/10^{-5}\text{cm}^{-2}\text{s}^{-1}$	k
17.0	2.0	3.1116	0.9881±0.0033
19.0	4.0	3.6703	0.9979±0.0032
21.0	6.0	3.2380	1.0009±0.0034
23.0	8.0	2.8648	1.0036±0.0023
25.0	10.0	2.2838	1.0032±0.0022

Table 4.2: average flux in Uranium core of radius R_u=15.0cm and eigenvalues for different paraffin shell width

flux at the boundary. When paraffin layer width keeps increasing, the system become super-critical, but graph (b) reveals that the change in eigenvalue is not significant from $w = 6.0cm$ to $w = 10.0cm$. Also, the flux graph shows that even though flux decreases slowly toward the outer sphere, the actual flux inside Uranium core is not very high. Hence, Despite that high scattering cross-section of paraffin can reduce leakage, the multiplication power of the system is compromised due to redistribution of flux toward peripheral non-fissile medium. For this reason, the critical mass of Uranium cannot be decreased to arbitrary value. The effectiveness of reflector is not necessarily proportional to the its thickness. Figure 4.7 shows the overall trend of eigenvalue when the reflector layer is thickened. the eigenvalue is expected to stop rising or even start to drop for further increase in w.

Comparing to the result in previous part, where we found R_c at 55% enrichment U sphere free of reflector to be 32.0cm. We see that this 6cm paraffin has so effectively reduced the mass of Uranium required for sustainable reaction.


x 10^−5
phi(r)
r/cm
w = 2.0cm
w = 10.0cm
w = 6.0cm


iter
w = 2.0cm
w = 10.0cm
w = 6.0cm

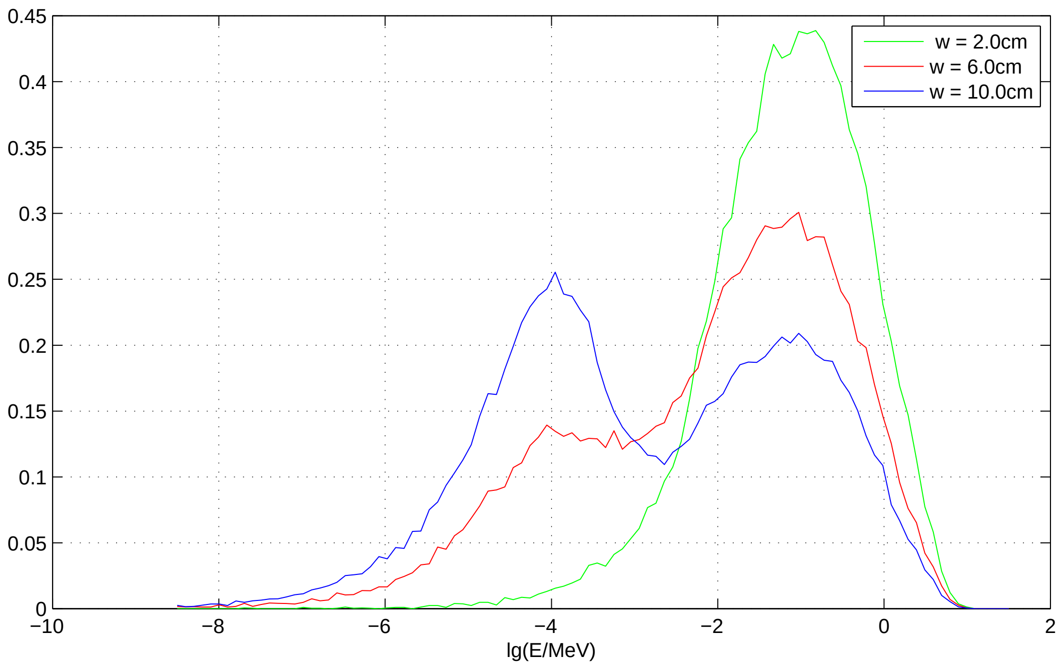


Figure 4.6: (a)Top: flux at three different reflector widths. (b)Middle: three eigenvalue curves. (c)Bottom: three energy spectra of the 60th batch

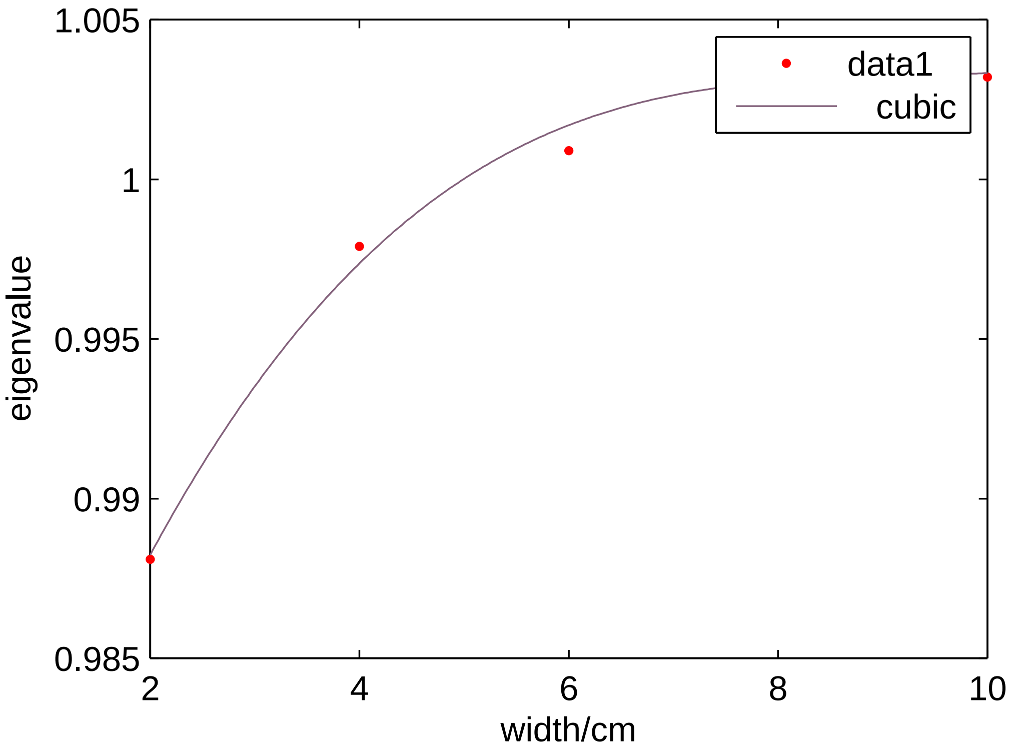


Figure 4.7: plot of eigenvalue against reflector thickness

Chapter 5

Conclusion

We have developed a working program that is capable of simulating the scalar flux and eigenvalue in a neutron transport problem for spherical reactor kernel. The performance of this program on three example problems provides results that agree with existing theory and past data. Through a direct comparison between Monte-Carlo simulation using fixed-energy cross-section at $E = 0.37$MeV and one-group deterministic calculation at the same group energy, we have confirmed the consistency between these two methods. By bringing in an energy dependent cross-section curve and a neutron fission spectrum, we have discovered a drop in the multiplication ability of the reactor core, which we explained by highlighting the larger scattering to fission probability at the new neutron peak energy. Even though this modified result is in less agreement with the existing data for critical mass of highly enriched Uranium, the MC method provides a way to examine the evolving energy distribution due to down-scattering. Determining critical condition tend to be tricky in MC method. However, using larger population and longer iteration cycles is seen to provide accurate enough estimation on the eigenvalue.

The second example confirmed a negative correlation between Uranium enrichment and critical mass for fast reactor. The energy spectrum is clearly shifted toward lower energy side as more U-238 is introduced, indicating more scattering in low-enriched Uranium. This shift in energy mainly appears as a skewing of the peak, while the width remains relatively stable, and negligibly little neutron reach energy outside 100eV $\sim$ 10MeV range. That is, all neutrons can be considered fast or intermediate neutrons. Extrapolating the rising curve of critical radius against decreasing enrichment, we predict that a limit exist below which critical reaction of unmoderated Uranium sphere is no longer possible.

The third example illustrate the effect of reflector in reactor design. Starting with the subcritical reactor with $R_u = 15.0$cm unmoderated Uranium core, we could reach a critical reaction when $w = 4.0$cm layer of paraffin is wrapped around it. Using the previous result for critical radius $r_c = 32.0$cm at 55% enrichment, the critical mass required will be 2570 kg, which is unrealistically large. However, a 6cm paraffin reflector can reduce critical mass to 265kg, nearly $\frac{1}{10}$ of its original value. This gives a simple but strong justification on such engineering design.

Bibliography

[1] International Atomic Energy Agency. Evaluated Nuclear Data File (ENDF). `https://www-nds.iaea.org/exfor/endf.htm`, 2016. [Online; accessed 2-April-2016].

[2] Japan Atomic Energy Agency. Nuclear data center. `http://wwwndc.jaea.go.jp/index.html`, 2016. [Online; accessed 2-April-2016].

[3] F. B. Brown, B. Nease, and J. Cheatham. Convergence testing for MCNP5 Monte Carlo eigenvalue calculations. Americal Nuclear Society, April 2007.

[4] Forrest B Brown. Fundamentals of monte carlo particle transport. *Los Alamos National Laboratory, LA-UR-05-4983*, 2005.

[5] Laurent Desorgher, E.O. Flückiger, M. Gurtner, M.R. Moser, and R. Bütikofer. Atmocosmics: A GEANT 4 code for computing the interaction of cosmic rays with the Earth's atmosphere. *International Journal of Modern Physics A*, 20(29):6802–6804, 2005.

[6] Roger Eckhardt. Stan Ulam, John Von Neumann, and the monte carlo method. *Los Alamos Science*, 15(131-136):30, 1987.

[7] Cornelius Joseph Everett and Edmond D Cashwell. Third Monte Carlo sampler. Revision and extension of samplers I and II. Technical report, Los Alamos National Lab., NM (USA), 1983.

[8] Donald Ervin Knuth. *The art of computer programming: sorting and searching*, volume 2. Pearson Education, 1998.

[9] Los Alamos National Laboratory. A General Monte Carlo N-Particle (MCNP) transport code. `https://laws.lanl.gov/vhosts/mcnp.lanl.gov/references.shtml#mcnp6_refs`, 2015. [Online; accessed 2-April-2016].

[10] Jaakko Leppänen. Serpent—a continuous-energy Monte Carlo reactor physics burnup calculation code. *VTT Technical Research Centre of Finland*, 4, 2013.

[11] Elmer E Lewis. *Fundamentals of nuclear reactor physics.* Academic Press, 2008.

[12] Elmer Eugene Lewis and Warren F Miller. *Computational methods of neutron transport.* John Wiley and Sons, Inc., New York, NY, 1984.

[13] David G Madland. Theory of neutron emission in fission. Technical report, Los Alamos National Lab., Theoretical Div., NM (United States), 1998.

[14] Robin Klein Meulekamp and Steven C van der Marck. Calculating the effective delayed neutron fraction with monte carlo. *Nuclear Science and Engineering*, 152(2):142–148, 2006.

[15] L.M. Petrie and N.F. Landers. An improved monte carlo criticality program. *NUREG/CR-0200*, 2, 1981.

[16] Shinzo Saito, Toshiyuki Tanaka, and Yukio Sudo. Design of high temperature engineering test reactor (HTTR). Technical report, Japan Atomic Energy Research Inst., 1994.

[17] G. D. Springgs, R. D. Busch, and J. M. Campbell. Calculation of the delayed neutron effectiveness factor using ratios of k-eigenvalues. *Annuals of Nuclear Energy*, 28:477–487, 2001.

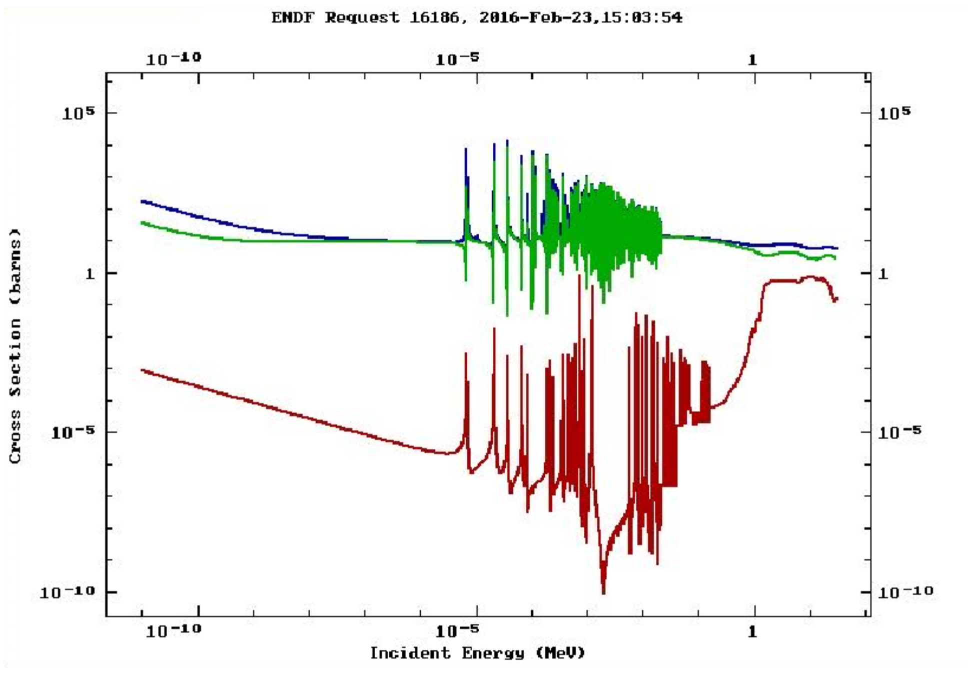
[18] J. Watterson. Lecture note on the fission spectrum, 2007.

[19] Christopher J Werner. Simulation of delayed neutrons using MCNP. *Progress in Nuclear Energy*, 41(1):385–389, 2002.

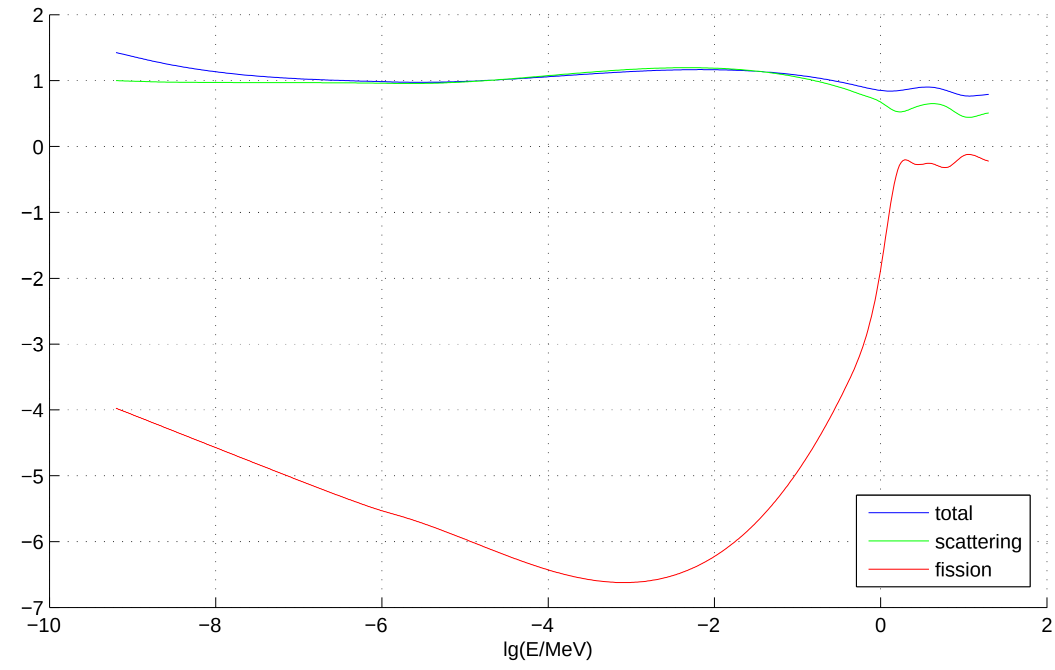
Appendices

Appendix A

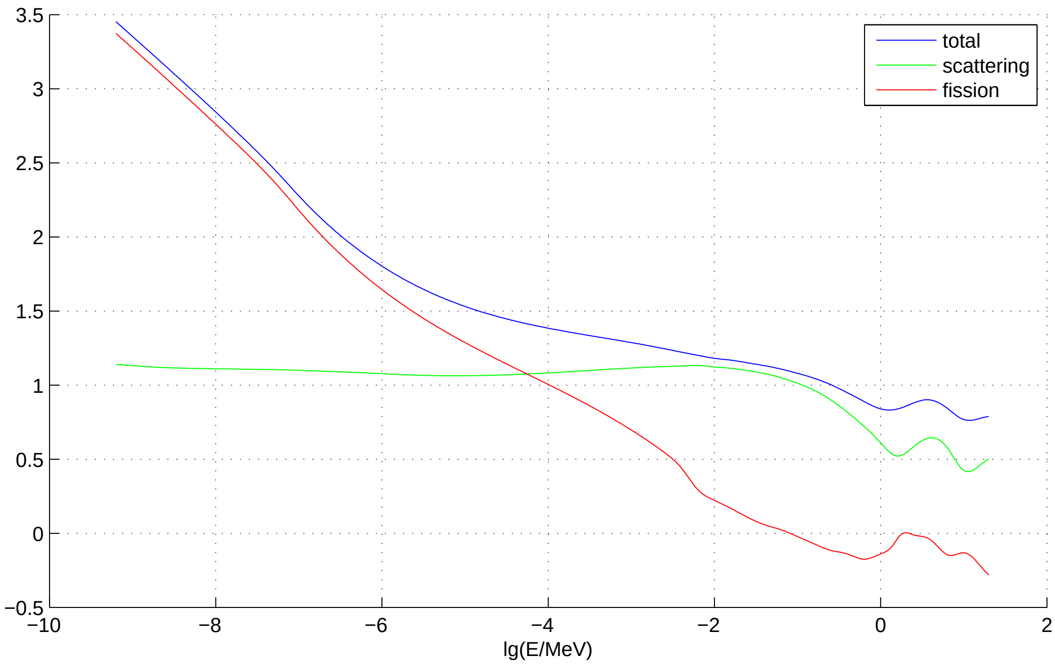
all interpolation graphs



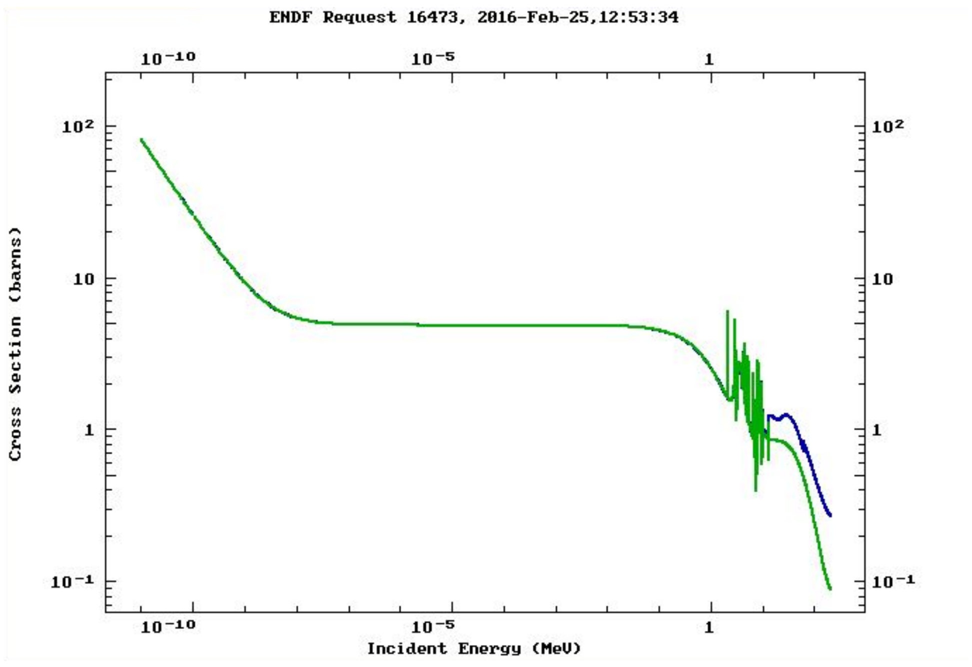
fission (red), scattering (green), and total (blue) cross-sections for U-238 based on ENDF/B-VII.1 experiment data.



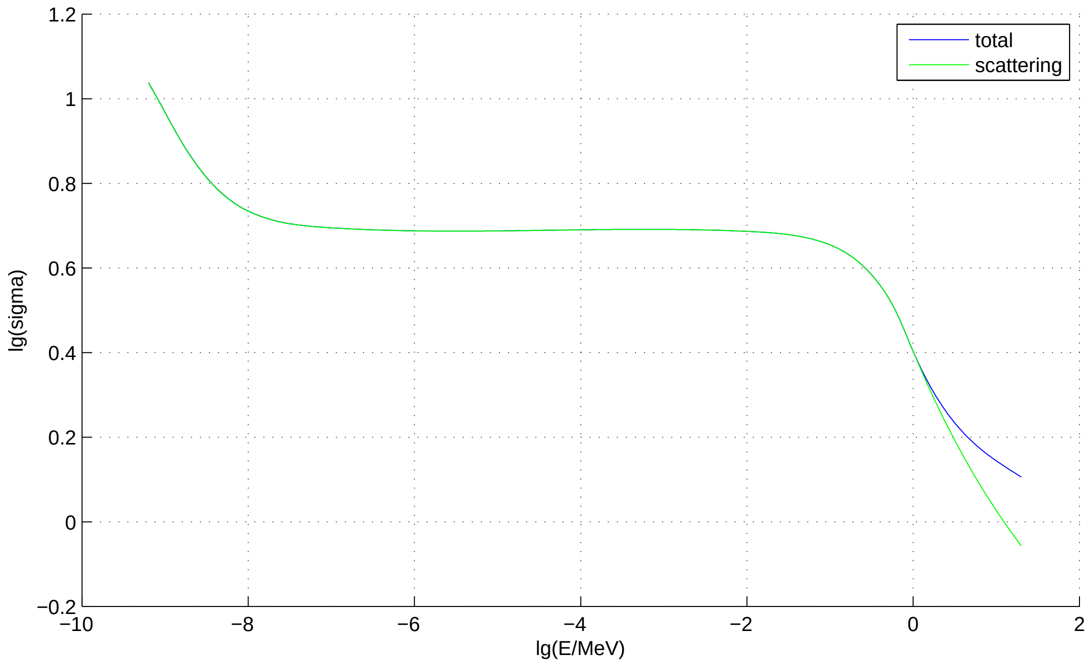
cubic spline interpolation for U-238 cross-sections



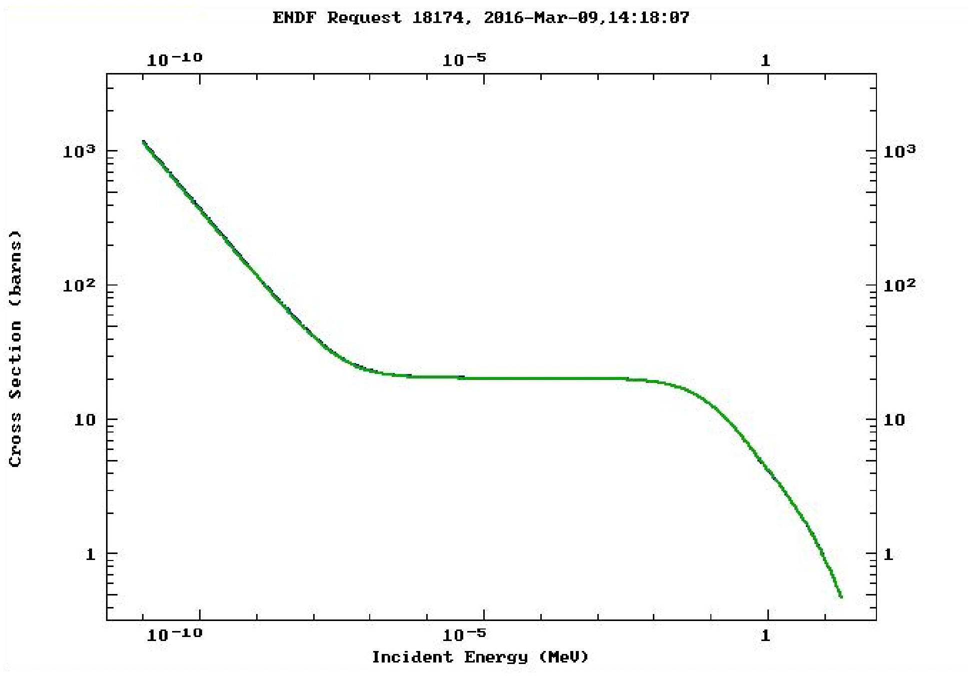
cubic spline interpolation for cross-sections of 60% enriched Uranium



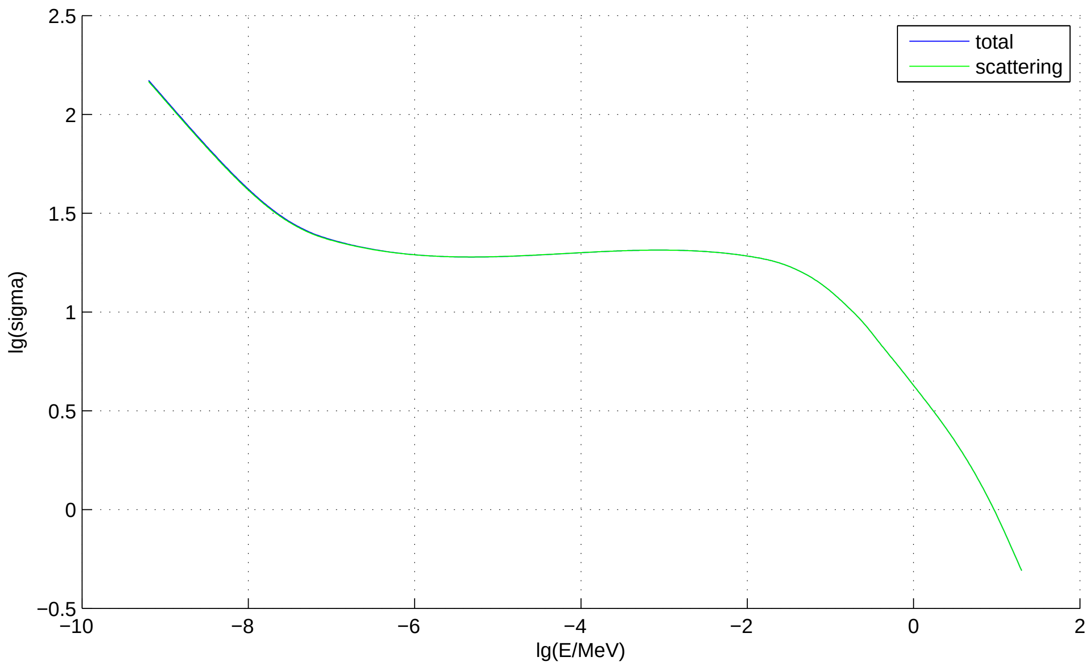
scattering (green), and total (blue) cross-sections for C-12 based on TENDL-2014 data.



cubic spline interpolation for C-12 cross-sections



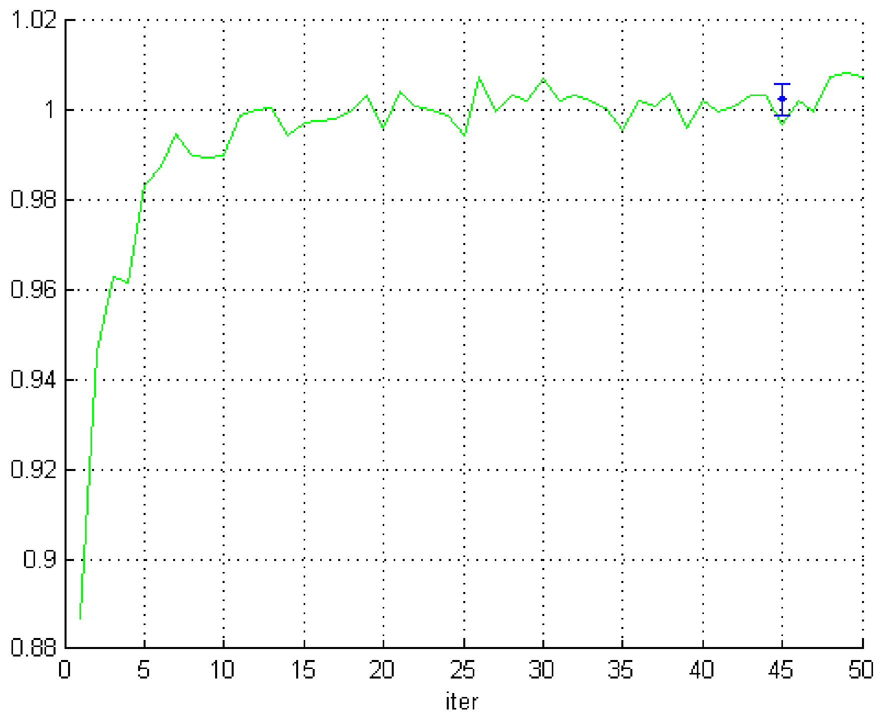
scattering (green), and total (blue) cross-sections for H-1 based on TENDL-2014 data.



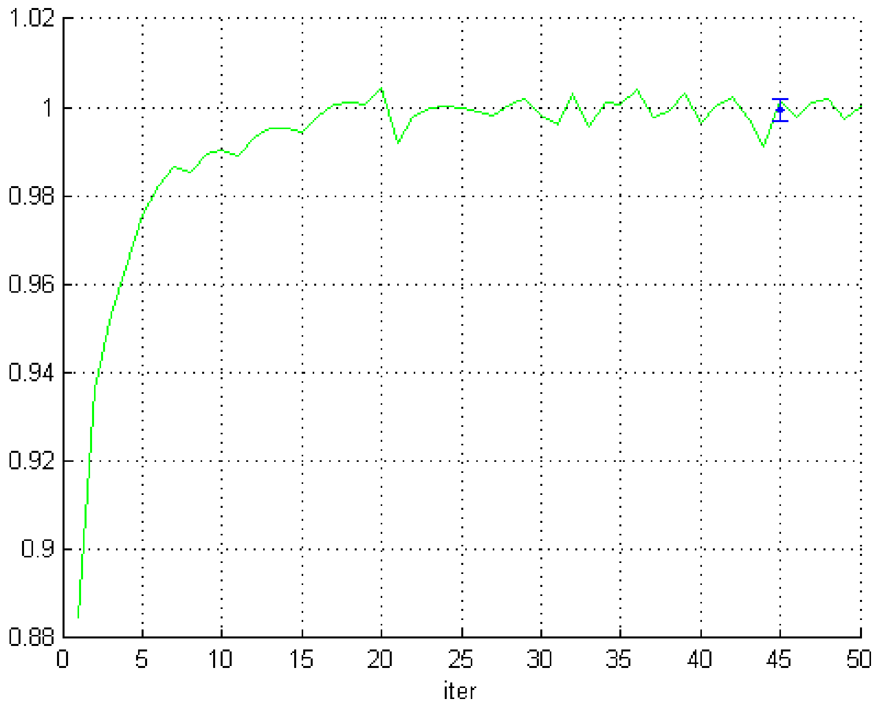
cubic spline interpolation for H-1 cross-sections

Appendix B

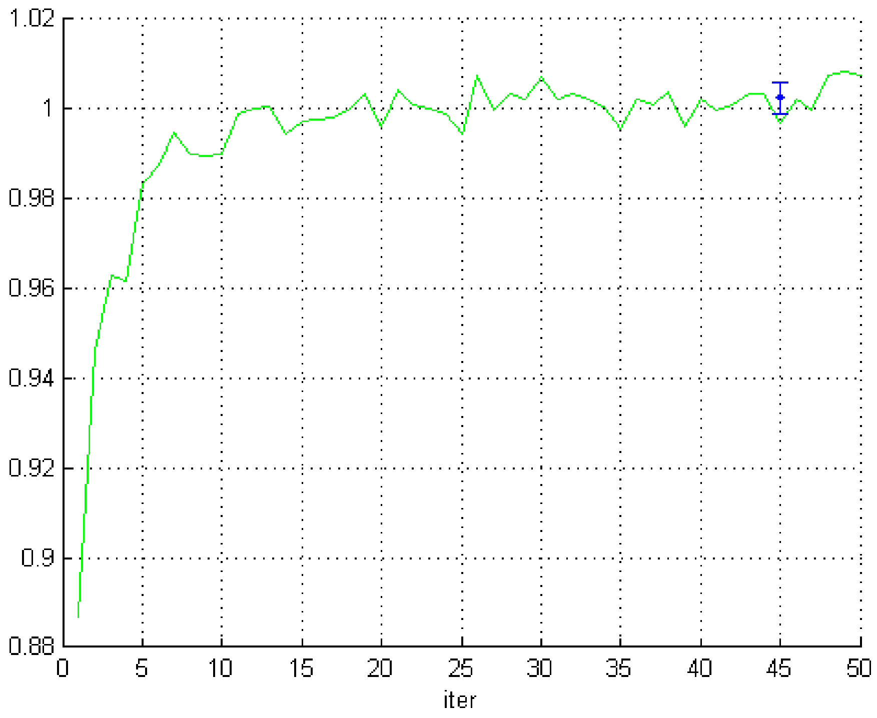
supplementary graph for enriched Uranium eigenvalues



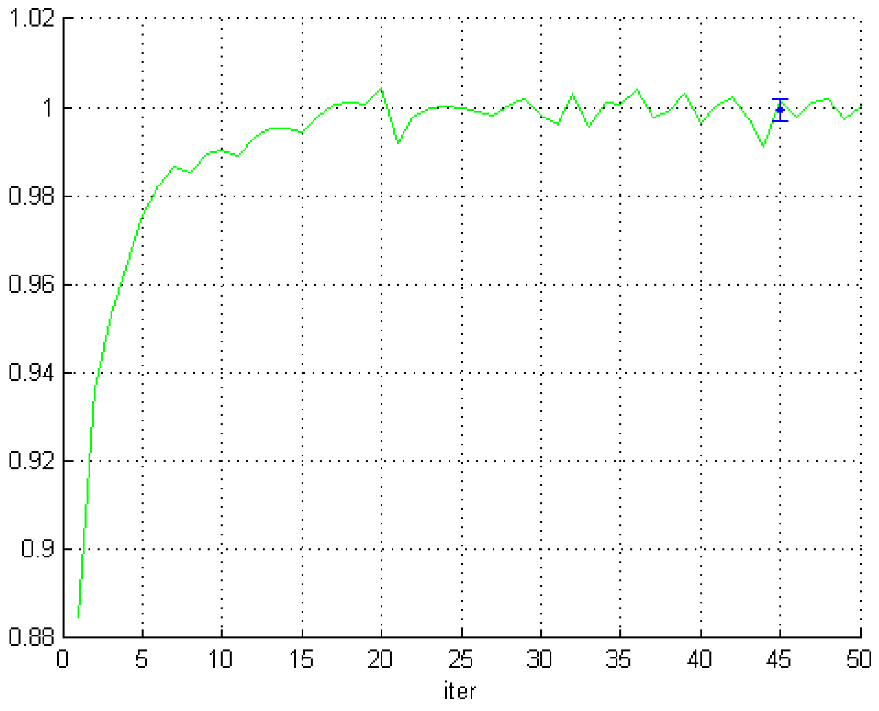
$e = 1.0,\ R_c = 10.8\text{cm},\ k = 1.0024 \pm 0.0035$



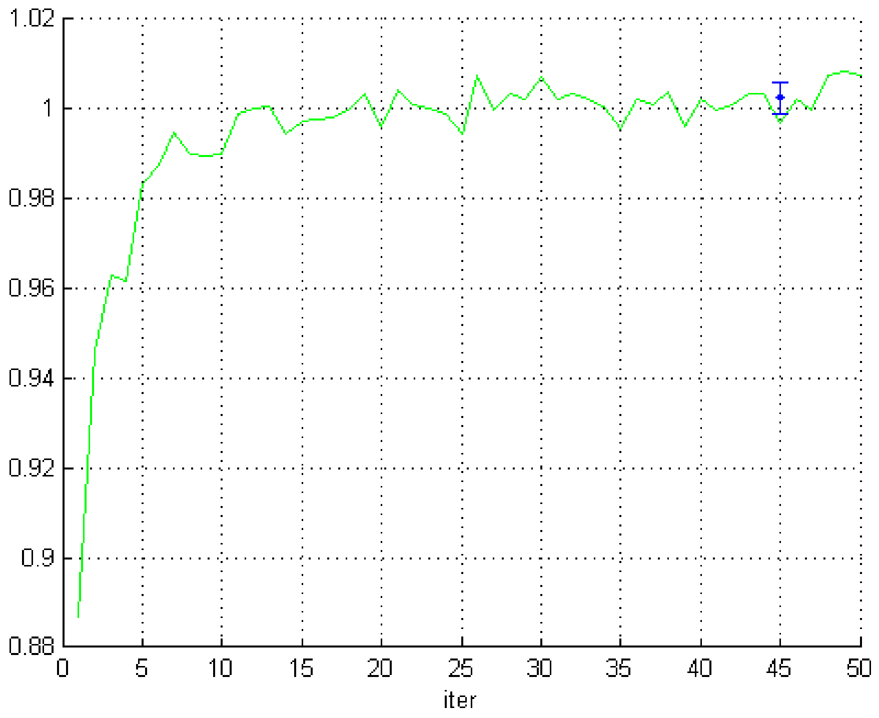
$e = 0.9,\ R_c = 12.4\text{cm},\ k = 0.9994 \pm 0.0029$



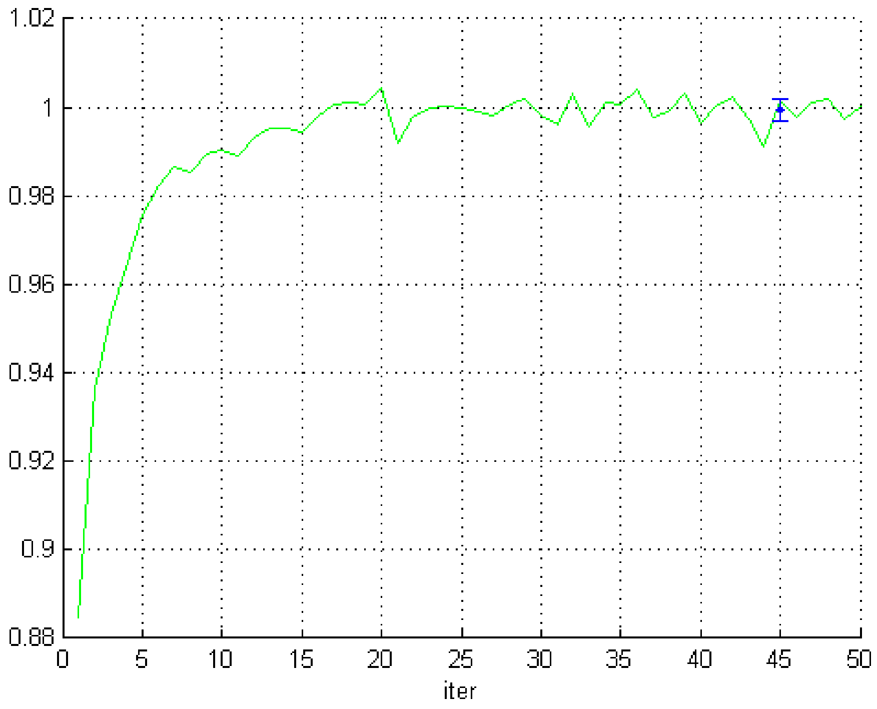
$e = 0.8,\ R_c = 14.9\text{cm},\ k = 1.0001 \pm 0.0034$



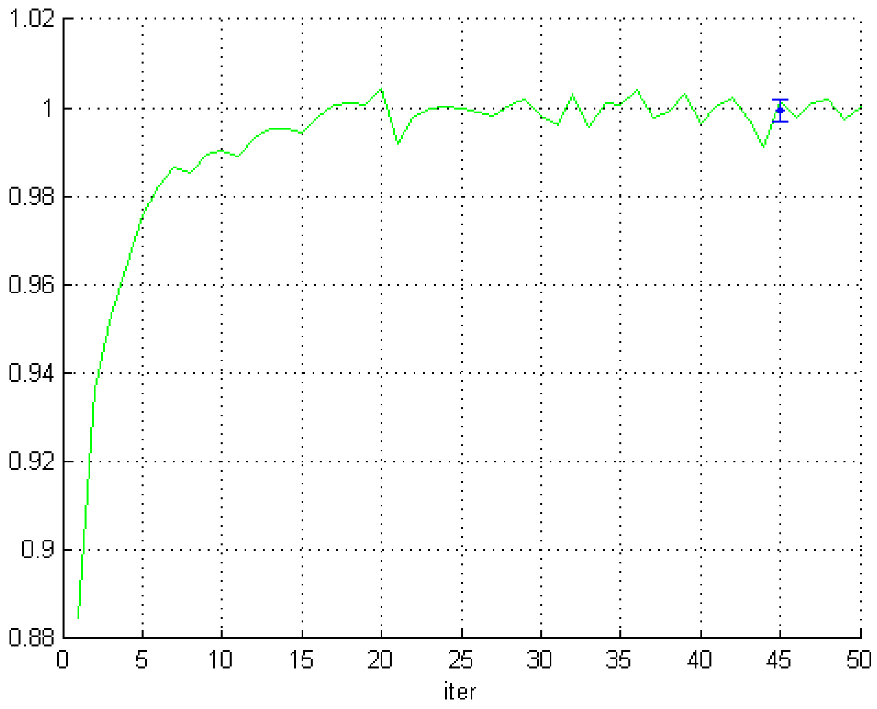
$e = 0.7,\ \ R_c = 19.0\text{cm},\ \ k = 1.0007 \pm 0.0028$



$e = 0.6,\ \ R_c = 25.2\text{cm},\ \ k = 1.0002 \pm 0.0023$



$e = 0.55,\ R_c = 32.0\text{cm},\ k = 0.9999 \pm 0.0026$



$e = 0.5,\ R_c = 50.0\text{cm},\ k = 1.0006 \pm 0.0031$

Appendix C

MATLAB code

```
%============ main ============
global Ru Rp; Ru=15.0; Rp=17.0;
global maxi; maxi = 30;  %dividing radial direction into 20 segments
H0 = 100000;
iter = 60;
ri = linspace(0,Rp,maxi+1);
V = 4/3*pi*Rp*Rp*Rp;
global Vi; Vi = 4/3* pi* ri.*ri.*ri;
Vi = Vi(2:end)-Vi(1:maxi);
global splineU235 splineU238 splineC12 splineH1;
MCtest2E_splineU235();
MCtest2E_splineU238();
MCtest2E_splineC12();
MCtest2E_splineH1();
eigen = zeros(1,iter);

rng(10);
global queueB phiB;

queueB = zeros(3,H0);
phiB = zeros(1,maxi);

%main program starts here
%0th batch

global lenB; lenB = 0;
for h=1:H0
    r=rand^(1/3)*Rp; %generate starting position
    mu = 2*rand -1; %generate direction cosine
    E = MCtest2E_spectrum;  %energies are actually lg(E)
    [sigmau, sigmasu, sigmafu] = MCtest2E_crossecu(E);
    [sigmap, sigmasp, sigmafp] = MCtest2E_crossecp(E);

    MCtest2E_walk(r,mu,E,sigmau,sigmap);
end
phitotB = sum(phiB)/V/H0;
phiB = phiB./Vi/H0;
```

```
figure;
plot(ri(2:end),phiB,'b-');pause(0.01);

%1st batch until nth batch
for n=1:iter

    phitotA=phitotB; phiB=zeros(1,maxi);
    lenA = lenB; lenB = 0;
    queueA = queueB(:,1:lenA); queueB = zeros(3,lenA*3);

    for h=1:lenA
        %retrieve r and mu
        r = queueA(1,h);
        mu = queueA(2,h);
        E = queueA(3,h);

        [sigmau, sigmasu, sigmafu] = MCtest2E_crossecu(E);
        [sigmap, sigmasp, sigmafp] = MCtest2E_crossecp(E);
        if r<Ru
            prob1 = sigmasu/sigmau;  %probability of scattering
            prob2 = (sigmasu+sigmafu)/sigmau;
        elseif r<Ru+Rp
            prob1 = sigmasp/sigmap;
            prob2 = (sigmasp+sigmafp)/sigmap;
        else
            fprintf('Err:radius out of reactor\n');
        end

        %decide collision type
        xi = rand;
        if xi<prob1  %scattering
            if r<Ru
                A = 235.0439/1.0087;
            else
                A = 14.016/1.0087;
            end
            chi = 2*rand -1;
            E = E+log10(A*A+2*A*chi+1)-2*log10(A+1);

            omega = 2*pi*rand;
            omega = cos(omega);
            chi = (1+ A*chi)/sqrt(A*A+2*A*chi+1);
            mu = mu*chi + sqrt((1-mu*mu)*(1-chi*chi))*omega;
            MCtest2E_walk(r,mu,E,sigmau,sigmap);
        elseif xi<prob2    %fission
            yield = 2+ ceil(0.43-rand);  %nu=2.43
            for p=1:yield
                mu = 2*rand -1;
                E = MCtest2E_spectrum;
                MCtest2E_walk(r,mu,E,sigmau,sigmap);
            end
        end
    end

    phitotB = sum(phiB)/V/H0;
    phiB = phiB./Vi/H0;
    eigen(n) = phitotB/phitotA;
    plot(ri(3:end),phiB(2:end),'b-');pause(0.01);
end
xlabel('r/cm'); ylabel('phi(r)');
%{
iu = ceil(Ru/Rp*maxi);
phiinu = sum(phiB(1:iu).*Vi(1:iu))/sum(Vi(1:iu));
%}

figure;
hold on;
plot(1:iter,eigen,'g-');
xlabel('iter');
grid;
```

```
%========== U235 spline ===========
%interpolation from E=-9.0 to E=1.2
function MCtest2E_splineU235()
    global splineU235;
    h = 0.2;
    n1 = int16((-7+9)/h)+1;
    n2 = int16((1.2+2.6)/h)+1;
    N = n1+n2;
    hlong = -2.6+7;
    m = int16(hlong/h)-1;


    sigTOT = [3.57226,3.47188,3.37072,3.26857,3.16582,...
              3.06139,2.95499,2.84614,2.73349,2.61384,...
              2.48880,...
              ...
              1.29549,1.25801,1.22252,...
              1.19162,1.17414,1.15135,1.13057,1.10688,...
              1.07737,1.04497,1.00122,0.94349,0.88391,...
              0.83236,0.83312,0.87812,0.89863,0.84237,...
              0.76524,0.77441];

    sigEL = [1.20507,1.19680,1.19114,1.18759,1.18528,...
             1.18320,1.18165,1.17945,1.17753,1.17449,...
             1.16963,...
             ...
             1.07417,1.07761,1.08478,...
             1.07176,1.06624,1.05345,1.04062,1.01535,...
             0.98268,0.94079,0.87365,0.78375,0.68378,...
             0.56189,0.51741,0.58380,0.64223,0.56554,...
             0.40574,0.45068];

    sigF =  [3.4949,3.3941,3.2925,3.1898,3.0867,...
             2.9824,2.8768,2.7689,2.6558,2.5329,...
             2.4005,...
             ...
             0.7662,0.6645,0.5166,...
             0.4437,0.3890,0.3292,0.2802,0.2456,...
             0.1996,0.1502,0.1064,0.0819,0.0466,...
             0.0791,0.0971,0.1009,0.0561,-0.0575...
```

```
                -0.1263,-0.2525];

    mat=4*h*diag(ones(1,N))+h*diag(ones(1,N-1),-1)+h*diag(ones(1,N-1),1);
    mat(1,1)=1; mat(1,2)=0; mat(end,end-1)=0; mat(end,end)=1;
    mat(n1,n1)=2*(hlong+h); mat(n1,n1+1)=hlong;
    mat(n1+1,n1+1)=2*(hlong+h); mat(n1+1,n1)=hlong;
    %display(mat);
    y=zeros(N,3);
    for i=2:N-1
        y(i,1)=3/h*(sigTOT(i+1)+sigTOT(i-1)-2*sigTOT(i));
        y(i,2)=3/h*(sigEL(i+1)+sigEL(i-1)-2*sigEL(i));
        y(i,3)=3/h*(sigF(i+1)+sigF(i-1)-2*sigF(i));
    end
    y(n1,1)=3/hlong*(sigTOT(n1+1)-sigTOT(n1))-3/h*(sigTOT(n1)-sigTOT(n1-1));
    y(n1+1,1)=3/h*(sigTOT(n1+2)-sigTOT(n1+1))-3/hlong*(sigTOT(n1+1)-sigTOT(n1));
    y(n1,2)=3/hlong*(sigEL(n1+1)-sigEL(n1))-3/h*(sigEL(n1)-sigEL(n1-1));
    y(n1+1,2)=3/h*(sigEL(n1+2)-sigEL(n1+1))-3/hlong*(sigEL(n1+1)-sigEL(n1));
    y(n1,3)=3/hlong*(sigF(n1+1)-sigF(n1))-3/h*(sigF(n1)-sigF(n1-1));
    y(n1+1,3)=3/h*(sigF(n1+2)-sigF(n1+1))-3/hlong*(sigF(n1+1)-sigF(n1));

    c = linsolve(mat,y);
    a = transpose([sigTOT;sigEL;sigF]);
    b = (a(2:end,:)-a(1:end-1,:))/h - (2*c(1:end-1,:)+c(2:end,:))*h/3;
    b(n1,:) = (a(n1+1,:)-a(n1,:))/hlong - (2*c(n1,:)+c(n1+1,:))*hlong/3;
    d = (c(2:end,:)-c(1:end-1,:))/3/h;
    d(n1,:) = d(n1,:)*h/hlong;


    a = [a(1:n1,:);zeros(m,3);a(n1+1:end-1,:)];
    b = [b(1:n1,:);zeros(m,3);b(n1+1:end,:)];
    c = [c(1:n1,:);zeros(m,3);c(n1+1:end-1,:)];
    d = [d(1:n1,:);zeros(m,3);d(n1+1:end,:)];


    for i=n1+1:n1+m

        for j = 1:3
            a(i,j)= a(i-1,j)+h*b(i-1,j)+h*h*c(i-1,j)+h^3*d(i-1,j);
            b(i,j)= b(i-1,j)+2*h*c(i-1,j)+3*h*h*d(i-1,j);
            c(i,j)= c(i-1,j)+3*h*d(i-1,j);
            d(i,j)= d(i-1,j);
        end
    end

    splineU235 = transpose([a,b,c,d]);

end
```

```
function [sigma,sigmas,sigmaf] = MCtest2E_crossecu(E)
    enrich = 0.55;  %the enrichment of U235

    global splineU235;
    global splineU238;
    N=52; h=0.2; Eleft=-9.0;

    i = floor((E-Eleft)/h)+1;
    if i<=0
        i=1;
    elseif i>=N
        i=N-1;
    end
    delE = E-(Eleft+h*(i-1));

    sigma235 = splineU235(1,i)+ splineU235(4,i)*delE+ splineU235(7,i)*delE^2+ splineU235(1
    sigmas235 = splineU235(2,i)+ splineU235(5,i)*delE+ splineU235(8,i)*delE^2+ splineU235(
    sigmaf235 = splineU235(3,i)+ splineU235(6,i)*delE+ splineU235(9,i)*delE^2+ splineU235(

    sigma238 = splineU238(1,i)+ splineU238(4,i)*delE+ splineU238(7,i)*delE^2+ splineU238(1
    sigmas238 = splineU238(2,i)+ splineU238(5,i)*delE+ splineU238(8,i)*delE^2+ splineU238(
    sigmaf238 = splineU238(3,i)+ splineU238(6,i)*delE+ splineU238(9,i)*delE^2+ splineU238(

    sigma = (10^(sigma235)*enrich+10^(sigma238)*(1-enrich))*0.0483278;
    sigmas = (10^(sigmas235)*enrich+10^(sigmas238)*(1-enrich))*0.0483278;
    sigmaf = (10^(sigmaf235)*enrich+10^(sigmaf238)*(1-enrich))*0.0483278;

end
```

```
function [sigma,sigmas,sigmaf] = MCtest2E_crossecp(E)
    global splineC12;
    global splineH1;

    N=52; h=0.2; Eleft=-9.0;

    i = floor((E-Eleft)/h)+1;
    if i<=0
        i=1;
    elseif i>=N
        i=N-1;
    end
    delE = E-(Eleft+h*(i-1));

    sigma12 = splineC12(1,i)+ splineC12(3,i)*delE+ splineC12(5,i)*delE^2+ splineC12(7,i)*d
    sigmas12 = splineC12(2,i)+ splineC12(4,i)*delE+ splineC12(6,i)*delE^2+ splineC12(8,i)*

    sigma1 = splineH1(1,i)+ splineH1(3,i)*delE+ splineH1(5,i)*delE^2+ splineH1(7,i)*delE^3
    sigmas1 = splineH1(2,i)+ splineH1(4,i)*delE+ splineH1(6,i)*delE^2+ splineH1(8,i)*delE^

    sigma = (10^(sigma12)+ 2*10^(sigma1))*0.03871;
    sigmas = (10^(sigmas12)+ 2*10^(sigmas1))*0.03871;
    sigmaf = 0;
end
```

```
function MCtest2E_walk(r,mu,E,sigmau,sigmap)
    global maxi Ru Rp queueB phiB lenB;

    L = -log(1-rand);   %optical path,
    d = r*sqrt(1-mu^2);      %perp distance of line of motion from centre
    if r<Ru
        l1 = -r*mu + sqrt(Ru^2-d^2);
        if L<l1*sigmau
            l = L/sigmau;
            sigmathere = sigmau;
        else
            l = l1 + (L-l1*sigmau)/sigmap;
            sigmathere = sigmap;
        end
    else
        if (mu<0 && d<Ru)      %going inward and cutting the inner shell
            l1 = -r*mu - sqrt(Ru^2-d^2);
            l2 = 2*sqrt(Ru^2-d^2);
            if L<l1*sigmap
                l = L/sigmap;
                sigmathere = sigmap;
            elseif L<(l1*sigmap+l2*sigmau)
                l = l1 + (L-l1*sigmap)/sigmau;
                sigmathere = sigmau;
            else
                l = l1 + l2 + (L-l1*sigmap-l2*sigmau)/sigmap;
                sigmathere = sigmap;
            end
        else
            l = L/sigmap;
            sigmathere = sigmap;
        end
    end

    rnext = sqrt(r*r+l*l+r*l*2*mu);

    if rnext<Rp
        lenB = lenB +1;
        delta = 1.0/rnext*(r+l*mu);
        mu = mu*delta + sqrt(1-mu*mu)*sqrt(1-delta*delta);
        r = rnext;

        queueB(1,lenB) = r;
        queueB(2,lenB) = mu;
        queueB(3,lenB) = E;
        i = ceil(r*maxi/Rp);
        phiB(i) = phiB(i)+1/sigmathere;
    end
end
```

```
function Elg = MCtest2E_spectrum()

    Elg = 15;
while Elg>10
    Elg0 = rand()*9-8;     %generate log(E) between -8 and 1
    E0 = 10^Elg0;
    fE = 0.4865*sinh(sqrt(E0*2))*exp(-E0);
    u = 0.4*rand();
    if u<fE
        Elg=Elg0;
    end
end
```

```

end
```