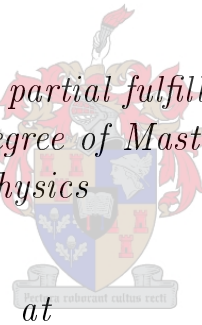


Monte Carlo Simulation of Direction Sensitive Antineutrino Detection

by

J.P. Blanckenberg

*Thesis presented in partial fulfillment of the
requirements for the degree of Masters of Nuclear
Physics*



at

Stellenbosch University

Physics

Natural Sciences

Supervisor: Dr. B.I.S. van der Ventel

Co-supervisor: Dr. F.D. Smit

Date: March 2010

Declaration

By Submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: March 2010

Abstract

Neutrino and antineutrino detection is a fairly new field of experimental physics, mostly due to the small interaction cross section of these particles. Most of the detectors in use today are huge detectors consisting of kilotons of scintillator material and large arrays of photomultiplier tubes. Direction sensitive antineutrino detection has however, not been done (at the time of writing of this thesis). In order to establish the feasibility of direction sensitive antineutrino detection, a Monte Carlo code, DSANDS, was written to simulate the detection process. This code focuses on the neutron and positron (the reaction products after capture on a proton) transport through scintillator media. The results are then used to determine the original direction of the antineutrino, in the same way that data from real detectors would be used, and to compare it with the known direction. Further investigation is also carried out into the required amount of statistics for accurate results in an experimental field where detection events are rare. Results show very good directional sensitivity of the detection method.

Opsomming

Neutrino en antineutrino meting is 'n relatief nuwe veld in eksperimentele fisika, hoofsaaklik as gevolg van die klein interaksie deursnee van hierdie deeltjies. Die meeste hedendaagse detektors is massiewe detektors met kilotonne sintilator materiaal en groot aantalle fotovermenigvuldiger buise. Tans is rigting sensitiewe antineutrino metings egter nog nie uit gevoer nie. 'n Monte Carlo kode, DSANDS, is geskryf om die meet proses te simuleer en sodoende die uitvoerbaarheid van rigting sensitiewe antineutrino metings vas te stel. Hierdie kode fokus op die beweging van neutrone en positrone (die reaksie produkte) deur die sintilator medium. Die resultate word dan gebruik om die oorspronklike rigting van die antineutrino te bepaal, soos met data van regte detektors gedoen sou word, en te vergelyk met die bekende oorspronklike rigting van die antineutrino. Verder word daar ook gekyk na die hoeveelheid statistiek wat nodig sal wees om akkurate resultate te kry in 'n veld waar metings baie skaars is. Die resultate wys baie goeie rigting sensitiwiteit van die meet metode.

Acknowledgements

I would like to thank the following people and organizations:

- My two supervisors, Dr. F.D. Smit and Dr. B.I.S. van der Ventel for their continuing support during my research. Due to the lack of available resources on the topic, I felt at times like Gloucester in William Shakespeare's play, King Lear, when he said "Tis the times' plague, when madmen lead the blind." Nevertheless, we all made it through.
- The Stichting EARTH for the opportunity to do this research that will have a direct impact on their project as well as for their financial support.
- iThemba LABS for the use of their facilities and for their funding in conjunction with the National Research Foundation.
- My Lord Jesus Christ who has given me the opportunities to discover more about the nature He created, and who has given me the strength and the capacity for that discovery.

Contents

Contents	i
List of Figures	iii
List of Tables	v
1 INTRODUCTION	1
1.1 Why Measure Antineutrinos?	1
1.2 Can Geoneutrinos Be Measured?	2
1.3 Method of Antineutrino Detection	2
1.4 Antineutrino Measurements In South Africa	5
1.5 Simulating Antineutrino Detection	5
2 ELEMENTARY PARTICLES	7
2.1 Standard Model	7
2.2 Fundamental Forces	9
2.3 Conservation Laws	10
2.4 Neutrino	11
3 THEORY	12
3.1 Source Of Geo-neutrinos	12
3.2 Antineutrino Detection	13
3.3 Bhabha Interaction Cross Section	13
3.4 Positron Energy Loss	17
4 SIMULATION	18
4.1 Program Flow	18
4.1.1 Neutron Transport	19
4.1.2 Positron Transport	20
4.2 Details of calculations	21
4.2.1 Initialization	22
4.2.2 Neutron	24
4.2.3 Positron	26
5 RESULTS	30
5.1 Positron Distribution	34
5.2 Positron And Neutron Together	37
5.3 Limited Neutron Position Resolution	42

6 CONCLUSION AND FUTURE PLANS	43
6.1 Error Due To Positron	43
6.2 Required Statistics	44
6.3 Future Plans	44
6.3.1 Current Status Of EARTH Project	45
A REACTOR SPECTRUM GRAPHS	47
B SIMULATION CODE	53
B.1 Main	53
B.2 MathFnc	60
B.3 InitializationFnc	63
B.4 WallsFnc	69
B.5 PositronScatteringFnc	72
B.6 NeutronFnc	80
Bibliography	83

List of Figures

1.1	Map of Earth with dots indicating positions of major man made nuclear fission reactors, taken from a talk by Fabio Mantovani which is published in Earth Moon and Planets based on information from [12].	3
3.1	The Feynman diagrams for the singlet (left) and triplet (right) interactions between a positron and a free electron.	14
4.1	Simple Overview of Simulation	18
4.2	Flow chart of neutron transport simulation	19
4.3	Flow chart of positron simulation, referred to as the Neutron subroutine	21
4.4	Scattering diagram of the initialization process in the lab system.	22
4.5	Figure showing the Feynman diagram of the (e^+, e^-) interaction where the initial momenta are denoted by \mathbf{p}_1 and \mathbf{p}_2 and the final momenta are indicated by \mathbf{p}'_1 and \mathbf{p}'_2	28
4.6	Triangle showing the conservation of momentum used to determine the angle Θ	29
5.1	Spectrum of antineutrinos coming from natural sources of uranium and thorium [22].	31
5.2	Spectrum of antineutrinos coming from nuclear fission reactors [22].	32
5.3	Diagram showing the angle Θ_o	32
5.4	Diagram showing the angle Θ_p	33
5.5	Graph of the angular distribution (with 0° being the direction of the antineutrino) of positrons for different energies ranging from 2 MeV to 9 MeV.	35
5.6	Graph of the radial distribution of the positron for different energies ranging from 2 MeV to 9 MeV.	36
5.7	Left: calculated distribution of positron annihilation coordinates in water projected onto a plane for ^{13}N source. Right: histogram of x coordinates from positron annihilation point distribution (Fig.7 in [23]).	36
5.8	Graph of the difference between Θ_o and Θ_p as a function of number of events. (Natural Spectrum)	37
5.9	Histogram of Θ_o calculated from single events. (Natural Spectrum)	38
5.10	Histogram of Θ_p calculated from single events. (Natural Spectrum)	39
5.11	Histogram of the difference between Θ_o and Θ_p , calculated from single events. (Natural Spectrum)	40

5.12	Θ_p vs. Number of events. (Natural Spectrum)	41
5.13	Θ_p vs. Number of events with error on the neutron position. (Natural Spectrum)	42
6.1	Simple model picture of what the GiZA detector will look like. . .	46
A.1	Graph of the difference between Θ_o and Θ_p as a function of number of events. (Reactor Spectrum)	47
A.2	Histogram of Θ_o calculated from single events. (Reactor Spectrum)	48
A.3	Histogram of Θ_p calculated from single events. (Reactor Spectrum)	49
A.4	Histogram of the difference between Θ_o and Θ_p , calculated from single events. (Reactor Spectrum)	50
A.5	Θ_p vs. Number of events. (Reactor Spectrum)	51
A.6	Θ_p vs. Number of events with error on the neutron position. (Reactor Spectrum)	52

List of Tables

2.1	Different quarks and their charges	8
6.1	Table showing the expected difference between using the positron and the actual origin of the reaction for the natural spectrum and the reactor spectrum.	43
6.2	Table showing the expected deviation of measured direction from actual antineutrino direction for the natural spectrum and the reactor spectrum.	44

Chapter 1 - INTRODUCTION

1.1 Why Measure Antineutrinos?

Information about the radioactive materials inside the earth is of special importance to geologists in the effort to discover the sources of heat emanating from the earth. It will also give more information regarding the heat sources that have kept the outer core of the earth hot enough to remain a fluid for this long. This can lead to further knowledge of the movement of heat and molten material at the various depths. Another question that is of interest to geologists is that of the possible existence of natural nuclear fission reactors deep within the earth. If the concentration of uranium (U) and thorium (Th) is high enough at a certain place, it has the potential to act as fuel for a natural reactor. The fission products are also radioactive, release energy, and are therefore also of interest. It is thought that such natural nuclear reactors may exist at the core-mantle boundary. If they are large enough, they may be visible on a tomographic image of radioactivity [1] inside the earth. The presence of natural nuclear fission reactors at the core-mantle boundary may have even been, according to one theory, responsible for the formation of the moon [2]. The big question is whether or not producing this 3-D map of radioactivity is possible.

The EARTH (Earth Antineutrino Tomography) project aims to produce a 3-D map of radioactivity in the earth [3]. The earth has a radius of approximately 6400 km though, whereas the deepest hole ever dug is approximately 12 km deep [4]. Beyond this depth however, the drilling equipment becomes too hot and breaks. Therefore obtaining information about the inside of the earth cannot be done by just taking samples. Some form of messenger from the deep that can carry the information, is needed. One way that information from deep inside the earth has been acquired is through the interpretation of seismic activity. This has made it possible to find out about the densities of different areas inside the earth, but cannot give any information regarding radioactive materials. One of the decay products of radioactive uranium (U), thorium (Th) and potassium (K) is an antineutrino¹. These are the perfect messengers since they are neutral, have almost no mass and are weakly interacting which allows them to travel through the earth virtually unhindered. If the antineutrinos from uranium and thorium² can be detected along with the direction they are coming from and what their energy is, it should be possible

¹Such antineutrinos originating from natural sources inside the earth are often referred to as geoneutrinos.

²Due to the Q value of the beta decay of ⁴⁰K, the maximum energy of an antineutrino resulting from this beta decay is too small for the detection process discussed in this thesis.

to create a tomographic image of the radioactive materials in the earth [5],[6]. Furthermore, the difference between the energy spectrum of antineutrinos from natural uranium and thorium and that of nuclear fission products would allow for differentiating of natural nuclear reactors, provided they are large enough. For a fully 3-D tomographic image, several points of detection spread over the earth for information from different angles will be required [6].

1.2 Can Geoneutrinos Be Measured?

Nunokawa *et al.* [7] showed how measuring the antineutrino flux at Kamioka and Gran Sasso can make it possible to distinguish between various models for the geophysical composition of the earth. According to them, an exposure time of more than a decade would be required in order to distinguish between the composition models. Similar calculations have been done by others [8],[9]. Their calculations however, were done for measuring just the antineutrino and neutrino flux, without their direction. Hochmuth [10] did Monte Carlo simulations with some directional sensitivity in order to distinguish between geophysical models of the earth. She determined that the antineutrino flux and direction would not be sufficient to distinguish between different models of the earth, but that the energy spectrum of antineutrinos would also be required. Hence there are three important properties of the antineutrinos that need to be measured. They are the flux, energy and direction. The latter of these three is the most challenging.

As mentioned already, the antineutrinos that are of interest come from U and Th in the earth, but those same elements also appear in man-made nuclear fission reactors. This means that nuclear reactors will add to the background in these detectors. The question then is whether or not there are areas on Earth where this background caused by nuclear reactors will be small enough so as not to overpower the antineutrinos coming from the earth. Fig. 1.1 shows a map of Earth with dots indicating the locations of nuclear reactors that would cause a background of antineutrinos. The map shows that there are still sufficient areas with no reactors. Due to the number of nuclear reactors in the USA, the far East and Europe, the most suitable locations for detecting antineutrinos from the earth would for instance be in the eastern region of the Atlantic Ocean, Australia and a place such as Hawaii [11].

1.3 Method of Antineutrino Detection

Since others have answered the questions regarding what information is required from the antineutrinos, and where on Earth would be suitable to detect these antineutrinos, the next challenge is how to obtain this information from antineutrinos. The same properties of antineutrinos that allow them to travel through the earth unhindered, also make them very hard to detect. We not only want to detect them, but we also want to know the direction from where they came, as well as their energy. Just the detection alone poses great challenges due to their small interaction cross section, but direction sensitive detection

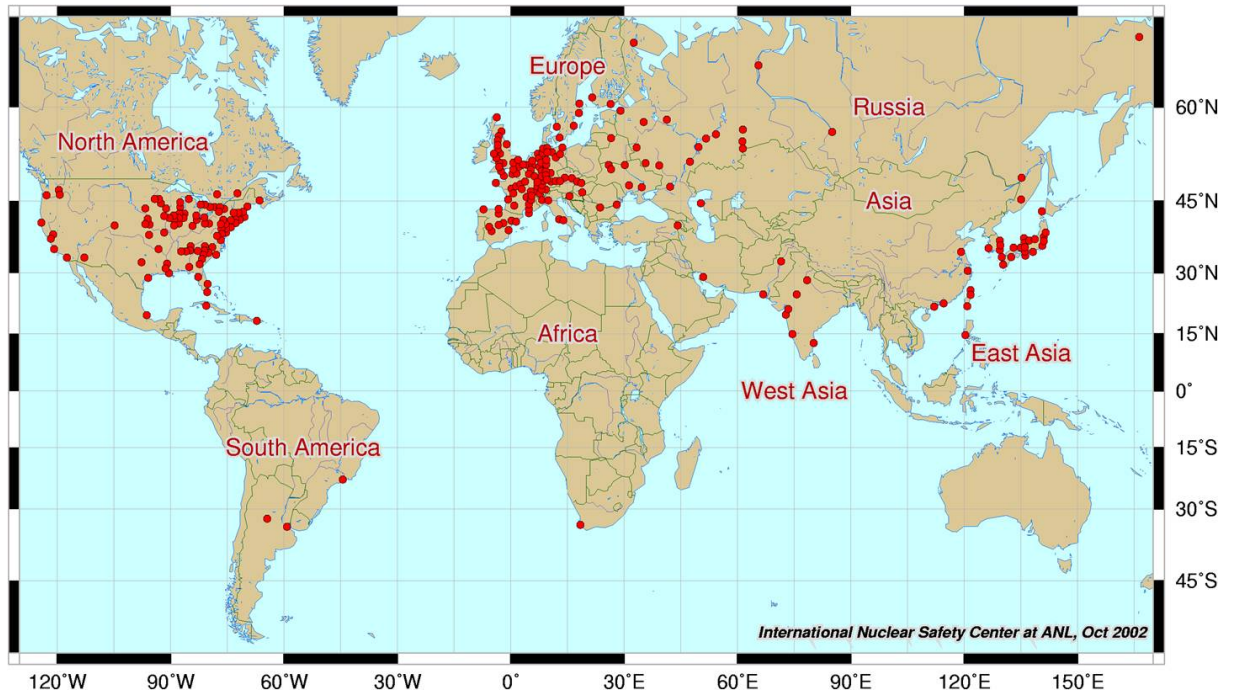


Figure 1.1: Map of Earth with dots indicating positions of major man made nuclear fission reactors, taken from a talk by Fabio Mantovani which is published in *Earth Moon and Planets* based on information from [12].

poses even greater challenges.

The most widely used method for detecting antineutrinos is through the inverse beta decay reaction:



In the detectors, the proton for the reaction is provided by the scintillation material and is therefore stationary. The positron has very little mass by comparison. It will therefore take most of the kinetic energy of the reaction. Since it is charged, it will deposit its energy into the scintillation material very quickly. The neutron is the most massive of the two reaction products and therefore takes almost all of the momentum of the initial particles. Since the proton was at rest, this implies that the neutron starts off in roughly the same direction as that of the antineutrino. The neutron will experience a few interactions with nuclei in the scintillator until it is attenuated to the point where it can be captured by some nucleus in the scintillator. An antineutrino event is therefore recognized by detecting a positron and a neutron close to each other in time and position.

Current antineutrino detectors are extremely large. The active neutrino target of the KamLAND detector has a diameter of 13 m and has 1200 m³ of liquid scintillator. It has 1325 17" aperture Photomultiplier Tubes (PMT) and

554 20" PMT's [13]. The CHOOZ detector contains 115 tons of scintillator and has 240 PMT's [14]. There is good reason for these large detectors namely that the cross section for an antineutrino event in the detector is so small, and that as many protons as possible is needed in order to get a reasonable detection rate. This large size has disadvantages too though. In these detectors, the distance between the reactions and the PMT's causes reduced accuracy and the fact that they consist of a single volume of scintillator surrounded by a large number of PMT's means that each event is seen by all the PMT's, making them very sensitive to background events. A further disadvantage of these detectors is that the neutron, after being attenuated, is captured by a hydrogen nucleus to form a deuteron and a 2.2 MeV γ . This γ will travel a few metres before being detected and this makes determining the position where the neutron was captured impossible.

An array of smaller detectors as proposed by De Meijer *et al.*[3] would have far fewer PMT's witnessing a single event and be closer to that event, which leads to greater accuracy as well as less dead time for the system of detectors as a whole [6]. For this reason the EARTH project detectors will be suitable for detecting these antineutrinos. The scintillator material in these detectors will be loaded with ^{10}B because ^{10}B can capture neutrons at a higher energy than the rest of scintillator. Hence the neutrons will undergo fewer interactions before being captured, and they will lose less directional information. The new ^{11}B is highly unstable and will decay to form ^7Li and an α particle which deposits its energy in the scintillator virtually immediately. This is what indicates where the neutron finally gets captured. The positron will lose its energy through bremsstrahlung and coulomb interactions with electrons in the scintillator until it is attenuated to a point where annihilation becomes likely. Because of the speed with which the positron loses its energy, it is expected that the total loss, caused by a lot of interactions will be detected as a single event. There are therefore two things to detect. A positron and a neutron. Drawing a line from where the positron is detected to where the neutron is detected will give some idea of the direction of the original antineutrino. In order to distinguish between the neutron and the positron, the detector must have good pulse shape discrimination (PSD). Differences between signal arrival time and pulse height at the different PMT's will be used to triangulate the positions of the events. This detection method poses several challenges and potential obstacles that are still unknown. One way to gain more information about these challenges is by doing simulations. Section 1.5 expands on this.

The design of the EARTH detectors to be placed underground has not yet been finalised. A detector for testing the properties of antineutrino detection, GiZA (Geoneutrinos in South Africa), will have the shape of a tetrahedron with 4 PMT's, one on each corner. Another possibility for the final shape is to have cylinders with PMT's at the ends. In either of these cases, a full detector will consist of many 'cells' of these shapes. The detectors will be placed deep underground to shield against muons and other cosmic particles. Due to their high energies though, they cannot be completely excluded which is why a cosmic veto will also be included to reduce background. There is also the possibility of tracking particles between cells in which case a positron may be detected in one cell, but the neutron from the same antineutrino event is

detected in a different cell. These will then be correlated.

1.4 Antineutrino Measurements In South Africa

There is close collaboration between the EARTH project and iThemba LABS, University of Stellenbosch (US), University of Cape Town (UCT) and University of the Western Cape (UWC) in South Africa since this is where the first prototype is planned to be tested. Parts and materials are being made in different places worldwide, but will come to South Africa to be tested. Different scintillation materials have already been experimented with in order to find a scintillator material that will give good timing as well as allow for the discrimination between positron events, neutron events and background. Thus far these experiments have been done on very small scale and no attempts at antineutrino measurements have been done yet. This is the purpose of GiZA, which will be placed at the Koeberg Power Station nuclear reactor in order to test the direction sensitive antineutrino detection.

1.5 Simulating Antineutrino Detection

The aim of our investigation is to write a program to simulate the inverse beta decay reaction (Eq. (1.1)). The program can then be used to investigate the feasibility of direction sensitive antineutrino detection.

Our program will also have the ability to simulate different shapes and sizes of detectors in order to see how that affects our efficiency.

In the detection process, the positron will be used to approximate the origin of the reaction, whereas the neutron will be used to find a second point which can be used to make a vector indicating the direction of the antineutrino.

- One of the most important questions is, "How big is the error we are making by using the positron as the origin?" Answering this question is the main objective of this investigation.
- Another important question is that of how much data is required to get a decent result. As stated already, antineutrinos rarely interact. This means that in the real detectors, collecting data will be a slow process and we need to know how much data we really need to get decent or really good results. The difference between 5000 and 50000 antineutrino events will be many years of waiting or a lot larger detector, and the extra counting time may not result in greatly improved results. Therefore this too is an important question.

Simulating this process poses several problems in the sense that this is a fairly new area. In the recent past, most simulation programs have worked with tabulated differential cross-sections for the required interactions and condensed simulations to reduce computation times [15]. With recent increases in computation power, those techniques are no longer necessary and direct simulations

can now be performed in a reasonable amount of time. Since the direct method has only recently become viable, there is very little documentation on it. There are other general purpose programs that can do what we are interested in, but it was decided that writing our own program would ensure complete understanding of every process in the program as well as complete flexibility of the program. Writing our own program also required a deep understanding of the physical processes that are being simulated as opposed to simply learning how to use an established program that takes care of all the physics by itself. Given the lack of freely available knowledge on this subject, it was very difficult to find sources of information on how to actually write our own program. It also means that aside from the general purpose programs, there are practically no sources for comparison. Therefore, even though the different sections of the program were tested fully and confirmed to be correct, the program as a whole could never be tested.

Chapter 2 - ELEMENTARY PARTICLES

This thesis is concerned with various particles, some having internal structure (neutron) and some having no internal structure (positron). Therefore an overview of our understanding of such particles is given here. For thousands of years man has tried to find the fundamental constituents of the universe. One of the earliest attempts at a group of fundamental constituents was fire, air, earth and water. Around the start of the twentieth century, physicists were on the right track (at least according to how we see the universe now). In those early years the electron was already known to exist and have charge. Ernest Rutherford's experiments showed that an atom (first thought to be the fundamental building blocks) consisted of a lot of empty space, with a dense nucleus in the centre and electrons moving around it at a relatively large distance. It was later determined that the nucleus consists of protons and neutrons. Protons have a positive charge with the same magnitude as the negative charge on an electron, but they have a much higher mass. Neutrons are neutral particles with a mass slightly higher than that of the proton. Because of the wave-particle duality of light, it be a particle and this is called the photon.

2.1 Standard Model

With the advent of accelerators it was possible to probe deeper into the mystery of the fundamental particles. Soon a myriad of particles apart from the neutron, proton, electron and photon were found. Clearly they were not as fundamental as previously believed. This called for a new theory. This theory is the Standard Model [16]. The Standard Model has proven to make very good predictions and so it is very well trusted, but it is also understood that it is not a theory of everything. Nevertheless, it is very useful within its boundaries. The Standard Model requires the existence of 6 quarks, 6 leptons and some force carrier particles to explain most of the observable universe (it does not explain gravity). The electron is the most well known lepton, the proton and neutron are made up of three quarks each and the photon is a force carrier particle.

The Standard Model also predicts an antiparticle for each particle. Antiparticles are identical to their corresponding particle except for the fact that they have opposite charge. The most common antiparticle we get is the positron or anti-electron. The positron is also a very important particle for this thesis. When a particle and (corresponding) antiparticle collide, they are annihilated and all that remains is pure energy according to Einstein's mass energy rela-

Charge	$\frac{2}{3}$	$-\frac{1}{3}$
	up	down
Quarks	top	bottom
	charm	strange

Table 2.1: Different quarks and their charges

tion, $E = mc^2$. Therefore when a positron and electron collide and annihilate, the energy is given off in the form of two γ rays each with an energy of 0.511 MeV, which is the mass of a single electron. Because of this annihilation, antimatter, at least in this part of the universe, does not last very long. The positrons in this thesis start off with relatively high energies though and this reduces the chances of annihilating. The energy needs to be lost through other forms before the positron can be annihilated.

As stated earlier, there are six kinds of quarks, also referred to as the six flavours of quarks. They are the up and down, the top and bottom and the charm and strange quarks. Of course there is an antiquark for each of them too. Quarks have charges of $\frac{2}{3}$ or $-\frac{1}{3}$ (see Table 2.1). With these fractional charges, different combinations can give different integer charges. Two up quarks and a down quark give a charge of $\frac{2}{3} + \frac{2}{3} - \frac{1}{3} = 1$. This combination is a proton. One up quark and two down quarks give a neutral charge and this combination is a neutron. Quarks always come in combinations. This is because the energy required to separate them from each other is large enough that it allows the formation of quark - antiquark pairs that then bind to the quarks that were originally bound together. It has already been explained that quarks come in six different flavours. They also come in three different colours, red, green and blue¹. This is important when looking at the different possible combinations of quarks. Any particle formed by quarks is called a hadron. They have integer charge and no colour charge. Hadrons can be separated into two groups: baryons and mesons. Baryons consist of three quarks, each of a different one of the three colours. The combination of these three colours gives white like in the combination of the colours of light. Of course the quarks are not really coloured. This property is just a simple way to represent the model in terms of something everyone is familiar with. Two examples of baryons are the proton and neutron. Mesons consist of quark- anti-quark pairs. This allows for the colour neutrality (red and anti-red gives white).

The next kind of particle is the lepton. There are six kinds of leptons. They are the electron, muon, tau and three neutrinos. There is one neutrino for each of the other three leptons. Neutrinos are neutral whereas the others are charged. Neutrinos will be discussed in more detail later.

¹Instead of green, yellow is sometimes used to match with the three basic colours in paint as opposed to light.

2.2 Fundamental Forces

These particles that we have found interact with each other in different ways in order to form everything in our universe. These interactions sometimes manifests like forces. There are only four types of interactions. The most familiar, but also least understood interaction, is gravity. The next most common one is the electromagnetic interaction and finally there are the strong force and the weak force which only act on a nuclear scale. For particles to interact with each other, something needs to be passed from the one to the other. All matter interacts with the exchange of particles called force carrier particle. Each interaction has a different force carrier particle (or more than one) associated with it.

The force carrier particle for the electromagnetic force is the photon (γ). It has no mass and travels at the speed of light, which also means that it has an infinite lifetime. Because of this, there is no range limit on the electromagnetic effect. It acts between charged particles like protons and electrons in an atom, but also over the great distances between stars.

There are other forces on the nuclear scale though. As stated earlier, quarks have colour charge and this gives rise to the strong interaction. The strong interaction is 137 times stronger (hence the name) than the electromagnetic interaction and its force carrier particle is the gluon. Gluons also have colour charge. Quarks that are close to each other interact by exchanging gluons. The farther apart the quarks are, the stronger the force is. This is converse to the electromagnetic force that gets weaker as the distance increases. As quarks exchange gluons, their own colour charge must change in order to satisfy conservation of colour charge. Because quarks have only one colour, the gluons must have two colour charges, a colour and an anti-colour charge. Inside a hadron, gluons are constantly being emitted, making the system very strongly bound. The strong force is so strong that when hadrons are close enough to each other, they will also be bound by the strong force. This is strong enough to overcome the electromagnetic force in a nucleus. The strong force is short ranged though so in larger nuclei, every nucleon does not 'see' every other nucleon via the strong force and the electromagnetic force becomes strong enough to break the nucleus apart. This is why heavier nuclei have more neutrons than protons. The neutrons enable the effect of the strong force to be larger, without increasing the electromagnetic force.

The weak interaction is responsible for any change of flavour of particles. So when a down quark in a neutron decays into an up quark to form a proton, that is the weak interaction changing the flavour. These kinds of flavour changes take place until the particles (quarks and leptons) are in the flavours with the smallest mass. The force carrier particle of the weak force are the W^+ , W^- and the Z . They are electrically positive, negative and neutral respectively. At very small ranges, the weak force and the electromagnetic force are equally strong (or weak), leading to their unification as the electroweak force. The W^+ , W^- and Z particles are quite massive though so this gives them a limited range as opposed to the massless photon.

The final interaction is gravity, but this cannot be explained using the

standard model. Since the effect of gravity is so weak however, it can be safely ignored in most particle physics problems.

2.3 Conservation Laws

Physicists have identified a few quantities that must always be conserved. These conservation laws govern all the interactions in the universe and without them everything would collapse.

- **Conservation of mass and energy:** Simply put, mass and energy can never be destroyed in a closed system. Mass can be converted to energy and vice versa, but it cannot be destroyed. In macroscopic systems energy may be lost to something like friction, but in those cases it is merely changed from mechanical energy to heat energy. The total mass and energy remains constant.
- **Conservation of momentum:** This is probably the most well known conservation law. In any closed system the total momentum must remain the same. As an example, if a particle at rest were to decay into two different particles that move away from each other, they will have to move in opposite directions and their total momentum have to add up to zero.
- **Conservation of electromagnetic charge:** This law is of special importance in decays or annihilation where particles change into other particles. A good example is β^- decay where a neutron, which is neutrally charged, decays to form a proton, electron and electron antineutrino. The charge before the decay was neutral and after the decay there is a proton and an electron, a positive charge and a negative charge which gives a total neutral charge once more.

$$n \rightarrow p + \beta^- + \bar{\nu}_e. \quad (2.1)$$

- **Conservation of lepton number:** The leptons come in three pairs, an electron, muon or tau, and its respective neutrino. The lepton number refers to the number of each one of the pairs. In the previous example of β^- decay, the lepton number for all three types was 0 before the decay. After the decay, an electron is created which means that electron lepton number is 1, but there is an electron antineutrino with electron lepton number of -1, bringing the total down to 0 once more.
- **Conservation of colour charge:** Whenever there is an interaction involving colour change, this law must be satisfied to. For instance, when a red quark emits a gluon, this gluon takes with it a colour so there must be a colour charge change in the quark. Assume the quark is blue after emitting the gluon. Then the gluon must carry two colour charges, red and anti-blue so that the total colour charge of the system remains red.

2.4 Neutrino

By studying radioactive decay, physicists found that some energy was ‘lost’. Since this would defy the conservation of mass and energy, there had to be some other explanation. The answer came in the form of the neutrino. The neutrino is a lepton with an extremely small (unknown) mass. As stated previously, there are three different types of neutrino, electron, muon and tau neutrinos. These have different masses, but their exact masses have not yet been determined. The small mass as well as the fact that neutrinos have no charge means that virtually the only interaction applicable to the neutrino is the weak interaction. Since the range of the weak interaction is quite small, and it is not very strong, neutrinos rarely interact with anything, making them (and their anti-particles) extremely difficult to detect.

The detection method considered in this thesis is through inverse beta decay. Conservation of lepton number shows that for the reaction to take place, it has to be an electron antineutrino that interacts with the proton, changing one of its up quarks into a down quark.

$$\bar{\nu}_e + p \rightarrow n + \beta^+. \quad (2.2)$$

Chapter 3 - THEORY

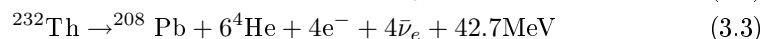
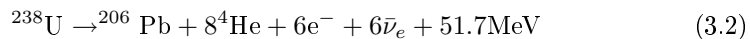
3.1 Source Of Geo-neutrinos

The antineutrinos (or geoneutrinos) that need to be measured in order to gain information about the distribution of radioactive materials inside the earth originate from the β^- decay of these radioactive materials. The reaction equation for β^- decay is given by:



Hence one of the neutrons in the nucleus decays to form a proton while the antineutrino and the electron (or β particle) are ejected away from the nucleus. A ^{232}Th will decay to form a ^{232}Pa . The number of nucleons remains the same, but since the number of protons increases by one, the element changes into the next element on the periodic table of elements. The β^- decay of each isotope releases a unique amount of energy and the antineutrino receives a specific amount of energy in the form of kinetic energy. In order to determine which nucleus decayed to form the antineutrino, it is only necessary to measure the energy of the antineutrino.

The three radioactive isotopes believed to be of greatest abundance in the earth are ^{238}U , ^{232}Th and ^{40}K . Each of these will give rise to a characteristic spectrum of antineutrinos when they decay. The energy of the antineutrino resulting from the decay of ^{40}K is too low for the current detection process to measure and it will therefore be ignored. The results of the decay chains of natural ^{238}U and ^{232}Th are shown in Eqn.(3.2) and (3.3). Note however, that Eqn.(3.2) and (3.3) only show the resultants after many decay's including β^- and α decays. The resultant energy in both cases gets carried away by the resultant particles in the form of kinetic energy so only part of that goes to the antineutrinos.



Antineutrinos are also emitted by the β^- decay of radioactive products of nuclear fission reactors. In this case, the main contributions are from the daughter nuclei resulting from the fission of ^{235}U , ^{238}U , ^{239}Pu and ^{241}Pu . This gives rise to a completely different spectrum of antineutrinos than the spectrum obtained from the natural sources. Furthermore any nuclear fission reactor that operates by the fission of these four isotopes will have a similar spectrum,

although slight variations are to be expected due to different percentages of the different daughter nuclei that actually give rise to the antineutrinos. These variations will however not be so much as to cause confusion with the spectrum of natural sources. This goes for natural nuclear fission reactors as well. As long as the fuel is the same as for the man made fission reactors, the antineutrino spectrum will be the same.

Provided that one can measure the energy of an antineutrino it should be possible to determine whether this antineutrino resulted from natural (non-fission) sources or from the fission fragments in nuclear fission reactors (natural or man made). Should there be natural nuclear reactors inside the earth, the added directional information of antineutrinos should aid in distinguishing them from the man made nuclear reactors.

3.2 Antineutrino Detection

The reaction used to detect antineutrinos, which we simulate, is the inverse beta decay:

$$\bar{\nu} + p \rightarrow n + \beta^+. \quad (3.4)$$

Since the cross section for antineutrino capture is extremely small, the simulation starts with the above reaction taking place. One of the important aspects about this reaction is the energy distribution between the positron and neutron. Given their great difference in mass, the neutron (having the greatest mass) will take away most of the momentum, and hence start off in roughly the same direction as the antineutrino. The positron on the other hand (having the smallest mass), takes away most of the energy of the antineutrino.

This inverse β -decay reaction has a threshold of 1.804 MeV [17]. Since the proton is at rest, the antineutrino needs to have an initial energy above this threshold or the reaction will not take place.

We should point out at this stage that all calculations are done in natural units. Hence

$$\hbar = c = 1. \quad (3.5)$$

3.3 Bhabha Interaction Cross Section

In order to simulate the transport of a positron through a scintillation medium, one needs to know the nature of its interactions with the particles in that medium. Figure (3.1) shows the Feynman diagrams of the singlet and triplet interactions between a positron and a free electron. The unpolarised scattering cross section for this interaction is given by Bhabha scattering [18].

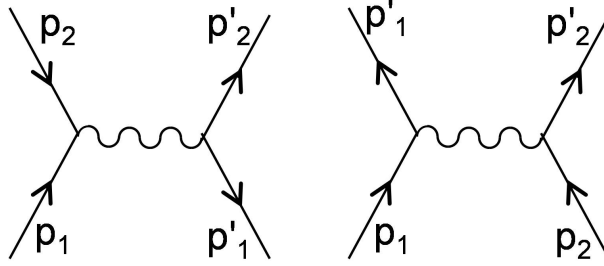


Figure 3.1: The Feynman diagrams for the singlet (left) and triplet (right) interactions between a positron and a free electron.

We define the following symbols:

$$\sigma_{inel}^+ = \text{cross section for inelastic scattering} \quad (3.6)$$

$$X_0 = \text{radiation length} \quad (3.7)$$

$$n = \text{electron density} \quad (3.8)$$

$$r_e = \text{classical electron radius} \quad (3.9)$$

$$m = \text{electron rest energy} \quad (3.10)$$

$$T = \text{kinetic energy of positron} \quad (3.11)$$

$$\epsilon = \frac{T'}{T} \quad (3.12)$$

$$\tau = \frac{T}{m} \quad (3.13)$$

$$y = \frac{1}{\tau + 2} \quad (3.14)$$

$$\beta^2 = \frac{\tau(\tau + 2)}{(\tau + 1)^2} \quad (3.15)$$

$$B_1 = 2 - y^2 \quad (3.16)$$

$$B_2 = (1 - 2y)(3 + y^2) \quad (3.17)$$

$$B_3 = B_4 + (1 - 2y)^2 \quad (3.18)$$

$$B_4 = (1 - 2y)^3 \quad (3.19)$$

$$T' = \text{kinetic energy of scattered electron.} \quad (3.20)$$

Using these symbols, the unpolarised scattering cross section for inelastic Bhabha scattering is given by [18]:

$$\frac{d\sigma_{inel}^+}{dT'} = \frac{X_0 n 2\pi r_e^2 m}{T^2} \left[\frac{1}{\epsilon} \left(\frac{1}{\epsilon\beta^2} - B_1 \right) + B_2 + \epsilon(\epsilon B_4 - B_3) \right]. \quad (3.21)$$

In order to find the total cross section, we need to integrate over all possible energies of the scattered electron. Usually one would integrate over all possible angles to find the total cross section, but there is a direct correlation between the angle and the energy so using the energy comes to the same result. The energy is used here because that is one of the known variables in the simulation.

$$\sigma_{inel}^+ = \int_{T_c}^T dT' \frac{d\sigma_{inel}^+}{dT'}. \quad (3.22)$$

3.3. BHABHA INTERACTION CROSS SECTION

15

Notice that T_c is the cutoff of the minimum energy of the scattered electron, but it is allowed to take all the energy. For completeness sake, we will evaluate this fairly straight forward integral.

$$\begin{aligned}\sigma_{inel}^+ &= \int_{T_c}^T dT' \frac{X_0 n 2\pi r_e^2 m}{T^2} \left[\frac{1}{\epsilon} \left(\frac{1}{\epsilon\beta^2} - B_1 \right) + B_2 + \epsilon(\epsilon B_4 - B_3) \right] \\ &= \frac{X_0 n 2\pi r_e^2 m}{T^2} \int_{T_c}^T dT' \left[\frac{1}{\epsilon} \left(\frac{1}{\epsilon\beta^2} - B_1 \right) + B_2 + \epsilon(\epsilon B_4 - B_3) \right] \quad (3.23)\end{aligned}$$

$$\int_{T_c}^T dT' \frac{1}{\epsilon^2 \beta^2} = \int_{T_c}^T dT' T^2 T'^{-2} \frac{1}{\beta^2} = -\frac{T^2}{\beta^2} \left(\frac{1}{T} - \frac{1}{T_c} \right) \quad (3.24)$$

$$\int_{T_c}^T dT' \frac{B_1}{\epsilon} = \int_{T_c}^T dT' \frac{T B_1}{T'} = T B_1 \ln \frac{T}{T_c} \quad (3.25)$$

$$\int_{T_c}^T dT' B_2 = B_2 (T - T_c) \quad (3.26)$$

$$\int_{T_c}^T dT' \epsilon B_3 = \frac{B_3}{T} \int_{T_c}^T dT' T' = \frac{B_3}{T} \frac{1}{2} (T^2 - T_c^2) \quad (3.27)$$

$$\int_{T_c}^T dT' \epsilon^2 B_4 = \frac{B_4}{T^2} \int_{T_c}^T dT' T'^2 = \frac{B_4}{T^2} \frac{1}{3} (T^3 - T_c^3). \quad (3.28)$$

$$(3.29)$$

Therefore

$$\begin{aligned}\sigma_{inel}^+ &= \frac{X_0 n 2\pi r_e^2 m}{T^2} \left[-\frac{T^2}{\beta^2} \left(\frac{1}{T} - \frac{1}{T_c} \right) - T B_1 \ln \frac{T}{T_c} + B_2 (T - T_c) \right. \\ &\quad \left. - \frac{B_3}{T} \frac{1}{2} (T^2 - T_c^2) + \frac{B_4}{T^2} \frac{1}{3} (T^3 - T_c^3) \right]. \quad (3.30)\end{aligned}$$

Now we define two more variables to simplify the above expression:

$$x = \frac{T_c}{T} \quad \text{and} \quad \gamma = \frac{E}{m}. \quad (3.31)$$

The following equations will be used to simplify the expression for the cross section.

$$\frac{T}{m} = \frac{E - m}{m} = \frac{E}{m} - 1 = \gamma - 1 \quad (3.32)$$

$$\frac{T - T_c}{T} = \frac{T}{T} - \frac{T_c}{T} = 1 - x \quad (3.33)$$

Similarly:

$$\frac{T^2 - T_c^2}{T^2} = 1 - x^2 \quad (3.34)$$

$$\frac{T^3 - T_c^3}{T^3} = 1 - x^3. \quad (3.35)$$

The cross section (Eq. (3.30)) now becomes:

$$\begin{aligned} \sigma_{inel}^+ &= \frac{X_0 n 2\pi r_e^2 m}{T^2} \left[-\frac{T^2}{\beta^2} \left(\frac{1}{T} - \frac{1}{T_c} \right) - T B_1 \ln \frac{T}{T_c} + B_2 (T - T_c) \right. \\ &\quad \left. - \frac{B_3}{T} \frac{1}{2} (T^2 - T_c^2) + \frac{B_4}{T^2} \frac{1}{3} (T^3 - T_c^3) \right] \quad (3.36) \end{aligned}$$

$$\begin{aligned} &= \frac{X_0 n 2\pi r_e^2}{T(\gamma - 1)} \left[-\frac{T^2}{\beta^2} \left(\frac{1}{T} - \frac{1}{T_c} \right) - T B_1 \ln \frac{T}{T_c} + B_2 (T - T_c) \right. \\ &\quad \left. - \frac{B_3}{T} \frac{1}{2} (T^2 - T_c^2) + \frac{B_4}{T^2} \frac{1}{3} (T^3 - T_c^3) \right] \quad (3.37) \end{aligned}$$

$$\begin{aligned} &= \frac{X_0 n 2\pi r_e^2}{\gamma - 1} \left[-\frac{T}{\beta^2} \left(\frac{1}{T} - \frac{1}{T_c} \right) - B_1 \ln \frac{T}{T_c} + B_2 \left(\frac{T - T_c}{T} \right) \right. \\ &\quad \left. - \frac{B_3}{2} \left(\frac{T^2 - T_c^2}{T^2} \right) + \frac{B_4}{3} \left(\frac{T^3 - T_c^3}{T^3} \right) \right] \quad (3.38) \end{aligned}$$

$$\begin{aligned} &= \frac{X_0 n 2\pi r_e^2}{\gamma - 1} \left[-\frac{1}{\beta^2} \left(\frac{T}{T} - \frac{T}{T_c} \right) + B_1 \ln x + B_2 (1 - x) - \frac{B_3}{2} (1 - x^2) + \frac{B_4}{3} (1 - x^3) \right] \\ &= \frac{X_0 n 2\pi r_e^2}{\gamma - 1} \left[\frac{1}{\beta^2} \left(\frac{1}{x} - 1 \right) + B_1 \ln x + B_2 (1 - x) - \frac{B_3}{2} (1 - x^2) + \frac{B_4}{3} (1 - x^3) \right] \quad (3.39) \end{aligned}$$

In the above derivation Eqs. (3.32) to (3.35) were used. Equation (3.39) is the final expression that will be used in the program for the scattering cross section of the positron through Bhabha scattering.

3.4 Positron Energy Loss

Two main forms of energy loss are simulated for the positron. The first is through the Coulomb interaction and it is given by the Bethe equation [19]¹:

$$\frac{dE}{dx} = \left(\frac{e^2}{4\pi\epsilon_0}\right)^2 \frac{4\pi NZ}{m_e v^2} \times \left(\ln\left(\frac{2m_e v^2}{I}\right) - \ln(1-v^2) - v^2 \right) \quad (3.40)$$

The second form of energy loss is bremsstrahlung and is given by [20]:

$$\frac{dE}{dx} = \frac{4NEZ(Z+1)e^4}{137m_e^2} \times \left(\ln\left(\frac{2E}{m_e}\right) - \frac{1}{3} \right). \quad (3.41)$$

The symbols are defined as follows.

$$E = \text{energy of positron} \quad (3.42)$$

$$e = \text{charge of electron} \quad (3.43)$$

$$\epsilon_0 = \text{permittivity of free space} \quad (3.44)$$

$$I = \text{mean ionization potential} \quad (3.45)$$

$$N = \text{number density of atoms in medium} \quad (3.46)$$

$$Z = \text{number density of protons in medium} \quad (3.47)$$

$$v = \text{velocity of positron} \quad (3.48)$$

$$x = \text{distance.} \quad (3.49)$$

Generally when one uses these equations, the energy loss is considered to be continuous as opposed to the true quantum nature of energy loss. For our purposes, the assumption is made that a positron will travel a certain distance and then scatter from an electron. The energy loss determined by the two equations is then transferred to the electron in an inelastic collision, changing the direction and energy of the positron. This assumption is not too far from the truth, however (especially in the case of the Coulomb interaction) since, even though the equations are continuous in nature, the true physical processes which they describe, are quantised.

¹The given version of this equation is for relativistic particles and in natural units where $c = 1$.

Chapter 4 - SIMULATION

4.1 Program Flow

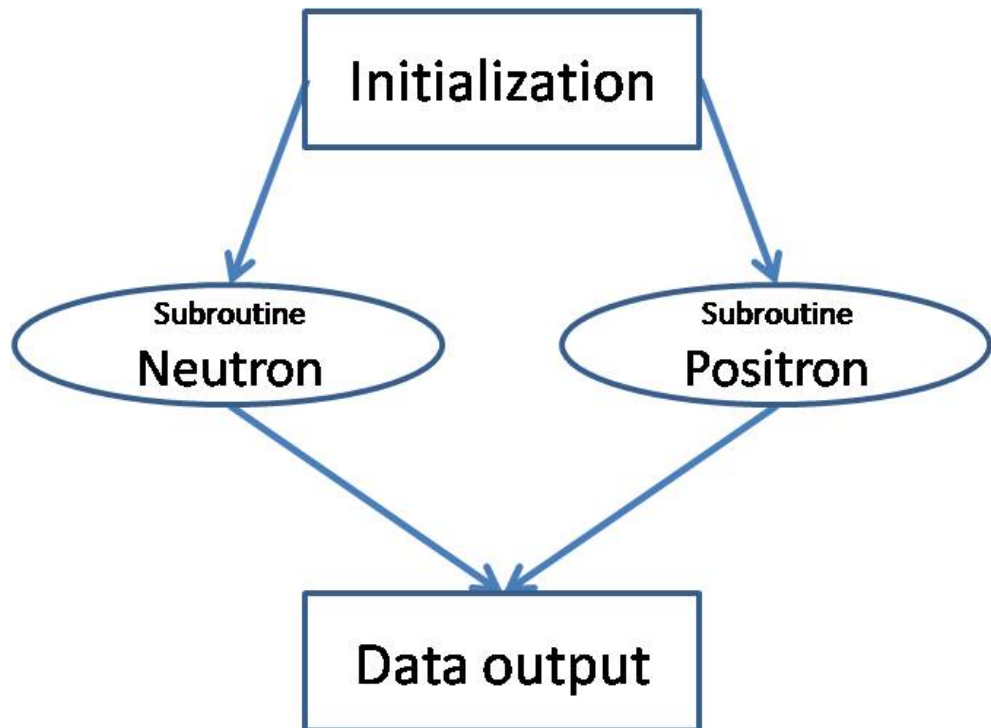


Figure 4.1: Simple Overview of Simulation

Figure 4.1 shows a simple overview of the simulation. In the initialization, the inverse beta decay is simulated and the momenta and positions (in the case where a finite detector is simulated) of the neutron and positron are simulated. This information is passed on to the subroutines **Neutron** and **Positron** respectively. The output of these subroutines are collected and saved in .csv files that can then be analyzed.

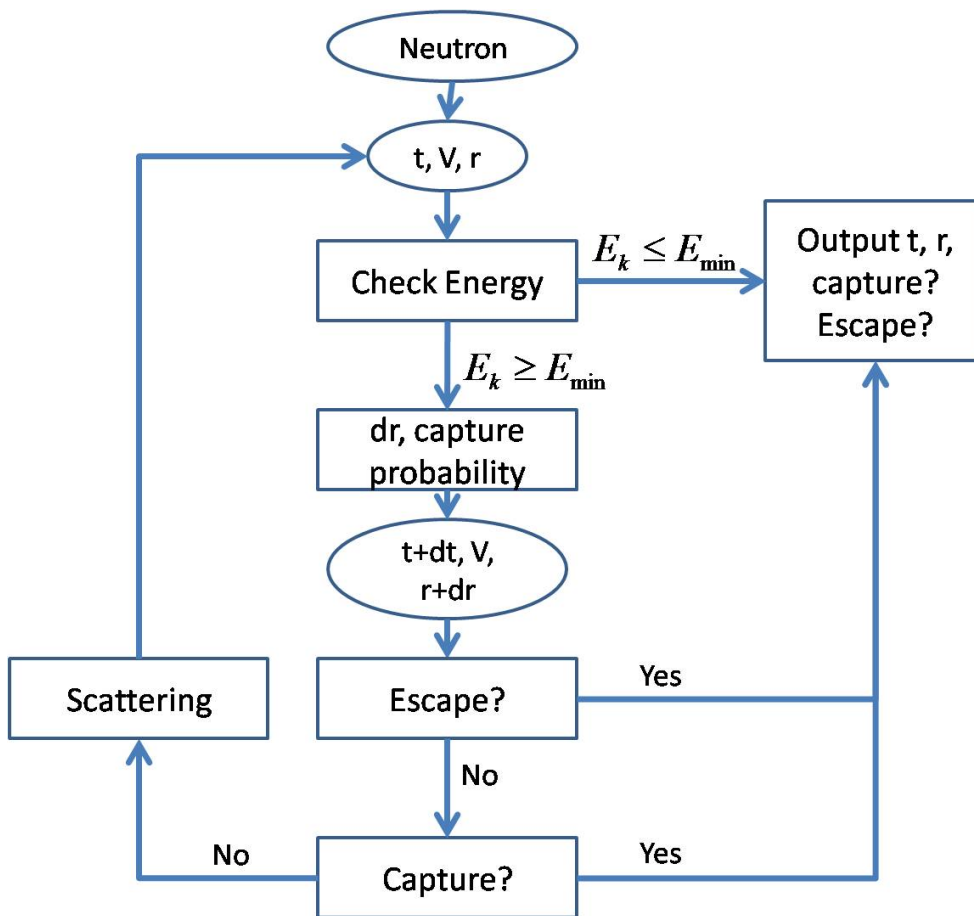


Figure 4.2: Flow chart of neutron transport simulation

4.1.1 Neutron Transport

Figure 4.2 shows a flow diagram of the neutron transport simulation or the **Neutron** subroutine. In the second oval from the top t , V and r represent the time, velocity (or momentum) and the position of the neutron. At first these are obtained from the initialization. The first step tests the kinetic energy of the neutron. If it is below a certain level, the neutron essentially disappears as far as detection is concerned. Thus, if the energy is too low, the **Neutron** subroutine ends. If the energy is higher than the threshold, it goes on to the next step.

In the next step, the neutron scattering cross section off H and B, as well as the probability of it being captured, are calculated. Both of these are functions of the energy. Using the scattering cross section and a random component, the distance traveled before the scattering or capture are calculated. Time and position are then updated.

Using the new position, the program checks to see if the neutron is still

inside the detector.¹ If the neutron is no longer inside the detector, we say it has escaped and the **Neutron** subroutine ends. If it is still inside, it goes on to the next step.

Now we use the capture probability calculated earlier, together with a random number, to see whether the neutron is captured or not. If it is captured, the subroutine ends. If not, it is scattered. It is important to note that this scattering process is non-relativistic, since it is a low energy neutron scattering off a stationary proton. The new direction of the neutron after the scattering has a random component and the magnitude of the velocity, which is dependent on the new direction, is calculated from two-body scattering.

The new time, position and velocity are then returned to the beginning of the loop. When the program ends, the time and position are given as output, as well as whether or not the neutron escaped or was captured or neither (in which case the energy was too low and it disappeared).

4.1.2 Positron Transport

Figure 4.3 is a flow diagram of the **Positron** subroutine. Bear in mind that since the positron is charged, and has a mass of almost 2000 times less than the neutron, their transport will differ greatly in key aspects. One of the most important aspects is that the positron, with an initial energy of no less than 1.8 MeV, is relativistic, whereas the neutron is not. Also, the positron is charged.

Similar to the **Neutron** subroutine, the **Positron** subroutine starts off with the time, momentum and position of the positron. Again a test is performed to see if the positron is energetic enough to be seen. If it is, it goes on to the next step.

The type of scattering the positron will undergo is electron-positron scattering and is called Bhabha scattering. This scattering cross section is calculated, and using a random component, the distance traveled is calculated. Since the positron loses energy through bremsstrahlung as well as due to Coulomb interactions, the energy loss is calculated using an expression for bremsstrahlung and the Bethe-Bloch equation respectively.

The next step is relativistic scattering. To simplify matters, all the energy lost (due to bremsstrahlung and Coulomb interactions) are simulated to be lost in the inelastic scattering from a free electron. After the scattering, the time, position and momentum are updated. As in the **Neutron** subroutine, the position of the positron is checked to see if it is still inside the detector. If not, the subroutine ends. If it is still inside, the loop starts again.

At the end of the subroutine, the time and whether or not the positron escaped is given as output, just as for the neutron, but because of the difference

¹This step applies only when simulating a finite detector and it is important when trying to optimise the shape and dimensions of the detector. We have built a cylindrical and a tetrahedral shape into the program, but other shapes can also be added.

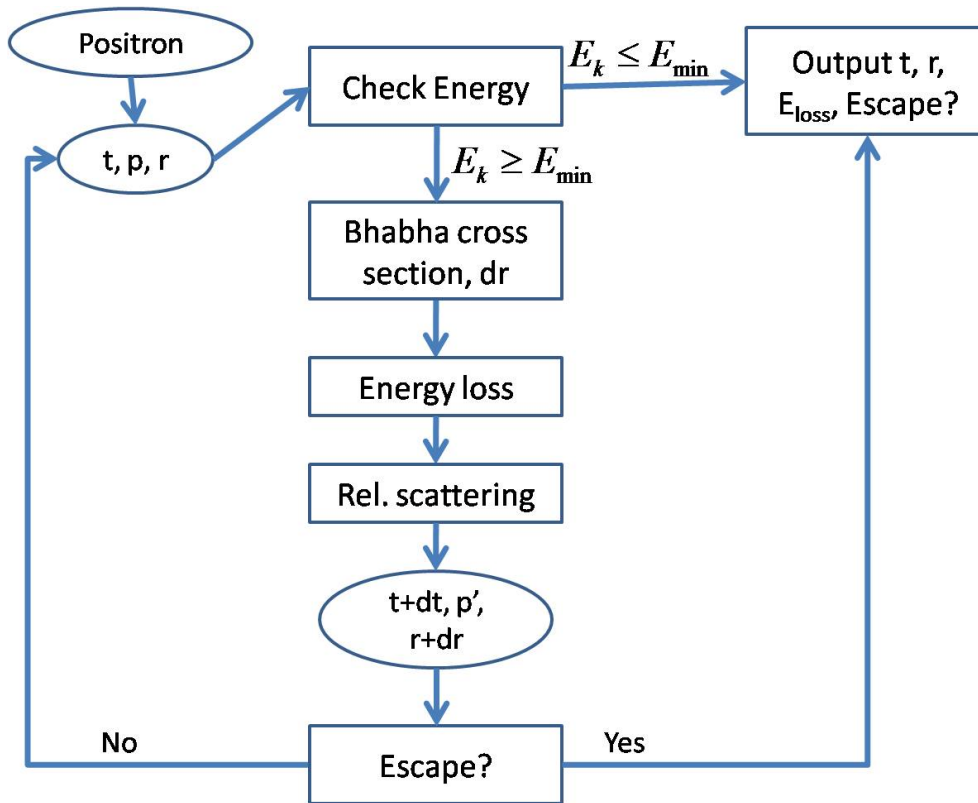


Figure 4.3: Flow chart of positron simulation, referred to as the **Neutron** subroutine

in the way these two particles are detected, the final position of the positron is not the only concern. What is important is where and how much energy is lost. There are a few ways this can be done. One of the first methods was to output the energy lost at each scattering in the data file, but this led to thousands of lines of data per positron. It was then decided that a more practical and more useful form of output would be the total amount of energy lost and the average position where this occurred.

4.2 Details of calculations

In this section we will go through all the mathematical details of the steps described in the previous section. It should be noted that there are many lines of code that are dedicated to ‘measurements’. These lines are used to find things like differences between certain times (like the total time of the neutron transport and the total time of the positron transport) or even just the progress of the simulation. Since they have nothing to do with physics or the simulation, they will not be mentioned here.

We will use a bold font (\mathbf{p}) to denote the vector of a momentum. The 4-momentum is denoted by capital letters (P^μ) and the magnitude of the 4-momentum is denoted by $|P^\mu|$.

4.2.1 Initialization

For the case where a finite detector is used, the first step would be to generate a random position inside the detector for the reaction to start at. Ensuring that each point in the detector is as likely as any other point is of great importance, but the details of that will not be discussed here since it differs from one shape to the next (for example a cube is much simpler than a sphere). Suffice to say that a random position is chosen, unless the detector is infinite, in which case the reaction starts at the origin $(0,0,0)$.

Conservation of momentum will be used to eventually find the 4-momenta of the neutron (P_3^μ) and the positron (P_4^μ) from the initial 4-momenta of the antineutrino (P_1^μ) and the proton (P_2^μ). These are all indicated on the scattering diagram in Fig. 4.4.

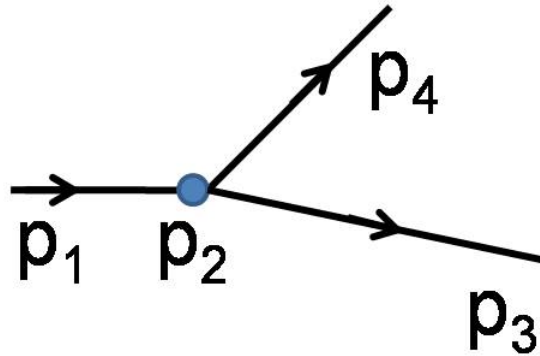


Figure 4.4: Scattering diagram of the initialization process in the lab system.

The first step is to determine the magnitude of the 4-momentum of the neutron.

$$P_1^\mu = (E, 0, 0, E) \quad (4.1)$$

$$P_2^\mu = (m_p, 0, 0, 0) \quad (4.2)$$

$$s = (P_1^\mu + P_2^\mu)^2 \quad (4.3)$$

$$E_{cm} = \sqrt{s} \quad (4.4)$$

$$E_3 = \frac{s - m_e^2 + m_n^2}{2E_{cm}} \quad (4.5)$$

$$a = (s - (m_n + m_e)^2)(2 - (m_n - m_e)^2) \quad (4.6)$$

$$|P_3| = \sqrt{\frac{a}{4s}}. \quad (4.7)$$

The total energy of the neutron, or the first component of P_3^μ , as well as the magnitude of the 4-momentum are now known. The calculations up to this point were taken from a Python code written by Fearick [21] for the propagation of a neutron in a scintillator. Next, a random direction on a sphere is generated in order to randomise the velocity of the neutron in the centre of mass system. Three random numbers are generated and denoted by r_1, r_2, r_3 . $sgn = 1$ if $r_3 < 0.5$, otherwise it is -1.

$$\phi = 2r_1\pi \quad (4.8)$$

$$z = 2r_2 - 1 \quad (4.9)$$

$$R = \sin \arccos z \quad (4.10)$$

$$v = (sgn \times R \cos \phi, R \sin \phi, z). \quad (4.11)$$

This velocity is multiplied by the magnitude of the 4-momentum to give us the neutron momentum.

$$P_{3_cms}^\mu = (E_3, v(0)|P_3|, v(1)|P_3|, v(2)|P_3|). \quad (4.12)$$

In order to convert to the lab system, a boost is applied in the z-direction, which is the direction of the incoming antineutrino.

$$\beta = \frac{P_1^3}{P_1^0 + |P_2|} \quad (4.13)$$

$$\gamma = \frac{P_1^0 + |P_2|}{E_{cm}} \quad (4.14)$$

$$v_0 = \gamma P_{3_cms}^0 + \gamma \beta P_{3_cms}^3 \quad (4.15)$$

$$v_3 = \gamma P_{3_cms}^3 + \gamma \beta P_{3_cms}^0 \quad (4.16)$$

$$P_3^\mu = (v_0, P_{3_cms}^1, P_{3_cms}^2, v_3). \quad (4.17)$$

The 4-momentum of the neutron in the lab system is now known. This means that three of the four 4-momenta are known so the 4-momentum of the positron can be found from conservation of momentum:

$$P_4^\mu = P_1^\mu + P_2^\mu - P_3^\mu. \quad (4.18)$$

The velocities of the particles can also be found by first calculating the kinetic energy and then the velocity from that. Note that the neutron is non-relativistic, but since the mass of the positron 0.511 MeV and its kinetic energy is of the order of 3 MeV, it will be relativistic.

Time is set to zero at this point so the time and the momenta and positions of both particles are now known and will be passed on to their respective subroutines.

4.2.2 Neutron

The neutron starts off with the time, momentum and position obtained in the **initialization** subroutine. The first non-trivial step in the subroutine is finding the displacement of the neutron and the probability of it being captured. Subscripts b, n and p relate to boron, neutron and proton respectively. They are also sometimes used together. The following symbols are defined:

$$v = \text{velocity of neutron} \quad (4.19)$$

$$\lambda = \text{mean free path} \quad (4.20)$$

$$\Sigma = \text{absolute cross section} \quad (4.21)$$

$$\rho = \text{density} \quad (4.22)$$

$$t = \text{triplet} \quad (4.23)$$

$$s = \text{singlet} \quad (4.24)$$

$$a = \text{scattering length} \quad (4.25)$$

$$r = \text{effective range} \quad (4.26)$$

$$r_r = \text{random number} \quad (4.27)$$

$$r_\phi = \text{random number} \quad (4.28)$$

$$r_z = \text{random number.} \quad (4.29)$$

In order to find the distance traveled (dr) and the probability of being captured (P_{cap}), the mean free path of the neutron in hydrogen and in boron needs to be calculated as follows.

$$v = \sqrt{\frac{T_3}{m_n}} \quad (4.30)$$

$$\lambda_b = \frac{1}{\rho_b \Sigma_b} \quad (4.31)$$

$$\lambda = \frac{1}{\frac{1}{\lambda_p} + \frac{1}{\lambda_b}} \quad (4.32)$$

$$\lambda_p = \frac{1}{\rho \Sigma_{np}} \quad (4.33)$$

$$E = \frac{1}{2} m_n v^2 \frac{m_p}{m_n + m_p} \quad (4.34)$$

$$m_{red} = \frac{m_p \times m_n}{m_n + m_p} \quad (4.35)$$

$$k = \sqrt{2m_{red} \frac{E}{(\hbar)^2}} \quad (4.36)$$

$$\Sigma_{np} = \frac{3}{4} \Sigma_t + \frac{1}{4} \Sigma_s \quad (4.37)$$

$$\Sigma_s = 4\pi \left(k^2 + \left(\frac{1}{a_s} + \frac{1}{2} r_s k^2 \right)^2 \right) \quad (4.38)$$

$$\Sigma_t = 4\pi \left(k^2 + \left(\frac{1}{a_t} + \frac{1}{2} r_t k^2 \right)^2 \right) \quad (4.39)$$

$$P_{cap} = \frac{\lambda}{\lambda_p} \quad (4.40)$$

$$dr = -\lambda \times \ln(r_r). \quad (4.41)$$

The distance traveled by the neutron until its next interaction and the probability of it being captured at that position are now known. The position is updated as well as the time (based on the velocity and the distance). The next step is to check that the neutron is still in the detector (when applicable). If so, a random number is generated between 0 and 1, and if this number is smaller than P_{cap} , the neutron is captured. If not, it goes on to (non-relativistic) 2-body scattering off a proton. This will change the direction and magnitude of the velocity of the neutron. This scattering is calculated as follows:

$$v_m = \frac{v}{\frac{m_p}{m_n} + 1} \quad (4.42)$$

$$v_{cm} = v - v_m \quad (4.43)$$

$$\phi = 2\pi(r_\phi) \quad (4.44)$$

$$z_i = (2 \times (r_z)) - 1 \quad (4.45)$$

$$\theta = \arccos z_i \quad (4.46)$$

$$z = z_i \times |v_{cm}| \quad (4.47)$$

$$x = |v_{cm}| \sin \theta \cos \phi \quad (4.48)$$

$$y = |v_{cm}| \sin \theta \sin \phi \quad (4.49)$$

$$v_{cm_new} = (x, y, z) \quad (4.50)$$

$$v_{new} = v_{cm_new} + v_m. \quad (4.51)$$

The new velocity of the neutron is now known. The velocity, time and position are passed back to the beginning of the loop to check the energy and so forth.

4.2.3 Positron

At the start, the only available information about the positron is its 4-momentum and position as given by the initialization. In order to simulate its propagation through the medium, the positron's velocity (\mathbf{v}) is required. This is obtained from its kinetic energy. Note that unlike the neutron, the positron is relativistic.

$$T = \sqrt{\mathbf{p}^2} \quad (4.52)$$

$$T = \frac{m_e c^2}{\sqrt{1 - \frac{v^2}{c^2}}} - m_e c^2 \quad (4.53)$$

$$\Rightarrow v = c \sqrt{1 - \left(\frac{T}{m_e c^2} + 1\right)^{-2}} \quad (4.54)$$

$$= \sqrt{1 - \frac{1}{\left(1 + \frac{T}{m_e}\right)^2}} \quad (4.55)$$

$$p_\perp = \sqrt{(p^1)^2 + (p^2)^2} \quad (4.56)$$

$$\theta = \arctan \frac{p_\perp}{p^3} \quad (4.57)$$

$$\theta_2 = \arccos \frac{p^3}{p^0} \quad (4.58)$$

$$\phi = \arctan \frac{p^2}{p^1} \quad (4.59)$$

$$\mathbf{v}(0) = v \cos \theta_2 \quad (4.60)$$

$$\mathbf{v}(1) = v \sin \theta \sin \phi \quad (4.61)$$

$$\mathbf{v}(2) = v \sin \theta \cos \phi. \quad (4.62)$$

Notice that there are two angles ‘ θ ’. This is because there are ambiguities in the way these two angles are calculated. They refer to the same angle, but in some cases might differ by a sign. We now have the energy, velocity, momentum and position of the positron. There is obviously redundancy since the former three can all be obtained from the 4-momentum, but further calculations are made faster by first calculating these explicitly.

Next the electron density is calculated. This depends on the constitution of the scintillator material and the calculation thereof depends on the form of the information available about the scintillator material. This is a fairly simple calculation though. The loop of the positron transport simulation is now started. The loop will end as soon as the positron energy drops below the cutoff value. The first step is to calculate the cross section for Bhabha scattering according to Eq. (3.39). All of the variables are calculated using the values previously calculated for the energy and velocity. A separate cross section is calculated for each of the three constituents of the scintillator (C,H,B). From this the mean free path is determined as follows.

$$n_{at} = Z \frac{N_{av} \rho}{A} \quad (4.63)$$

$$\lambda = \frac{1}{(\sigma_C \times n_{at,C}) + (\sigma_H \times n_{at,H}) + (\sigma_B \times n_{at,B})} \quad (4.64)$$

where N_{av} is Avogadro’s number and A is the proton number. Using the mean free path, a distance traveled (dr) is simulated by:

$$dr = -\lambda \times \ln(r_p) \quad (4.65)$$

where r_p is a random number. This distance is used to update the position of the positron. Notice that the negative sign is due to the fact that the random number is between 0 and 1, which means that the \ln of that will be between $-\infty$ and 0. With the position updated, a test is done to see if the positron is still inside the detector as well as other things regarding its trajectory and displacement, all of which are specific to the required measurements.

As mentioned earlier, two mechanisms of energy loss (Eqs. (3.40) and (3.41)) are simulated. Since the distance traveled is known, these two equations are simply used to determine how much energy is lost by each process and the results are added together. The next step is to transfer this energy by scattering off an electron. The easy part is to find the new energy of the positron by subtracting that amount which will be lost from the original energy. The more difficult part is finding the angle of the positron after the scattering. This angle is determined by the kinematics. Fig. 4.5 shows the Feynman diagram for the simple scattering interaction.² Particle 1 is the positron and particle 2 is the electron, since the following algorithm will work for any two particles and not just for the simplified case of two particles with the same mass. It was tested using different particles (from electrons to neutrons) and energies and compared to the results of ‘Kin2Body’, a widely accepted program for doing

²The kinematics for the singlet and triplet are the same so only one is shown here.

two body scattering problems. The results obtained from this subroutine were exactly the same as that of 'Kin2Body' to the greatest accuracy available. In order to find the scattering angle, the final momenta of both particles will be used so they have to be calculated first.

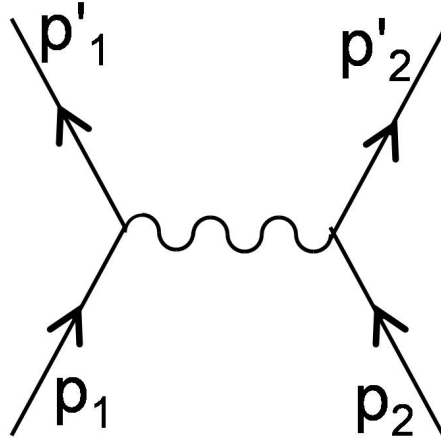


Figure 4.5: Figure showing the Feynman diagram of the (e^+, e^-) interaction where the initial momenta are denoted by \mathbf{p}_1 and \mathbf{p}_2 and the final momenta are indicated by \mathbf{p}'_1 and \mathbf{p}'_2 .

$$E_1 = p_1(0) \quad (4.66)$$

$$E'_2 = m_2 + E_{loss} \quad (4.67)$$

$$E'_1 = E_1 - E_{loss} \quad (4.68)$$

$$p_2'^2 = E_{loss}^2 + 2m_2 E_{loss} \quad (4.69)$$

$$p_1'^2 = E_1'^2 - m_1^2 \quad (4.70)$$

$$p_1^2 = E_1^2 - m_1^2. \quad (4.71)$$

Since the electron is initially at rest, conservation of momentum requires that the sum of the momenta of the two particles after the scattering have to add up to the momentum of the first particle before the scattering. Thus, these three vectors form a triangle (see Fig. 4.6) and using simple geometry and the cosine rule, the angle that we are interested in, namely the angle between the p_1 and p'_1 , can be found.

$$\cos \Theta = \frac{\mathbf{p}_1^2 + \mathbf{p}_1'^2 - \mathbf{p}_2'^2}{2\sqrt{\mathbf{p}_1^2}\sqrt{\mathbf{p}_1'^2}}. \quad (4.72)$$

Thus the scattering angle is now known. The following general rotation matrix is used to rotate any vector through an angle θ around the axis (x, y, z) .

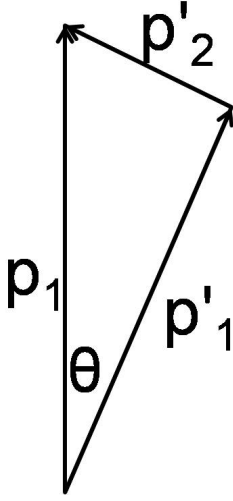


Figure 4.6: Triangle showing the conservation of momentum used to determine the angle Θ .

$$\begin{bmatrix} 1 + (1 - \cos(\theta))(x^2 - 1) & -z \sin(\theta) + (1 - \cos(\theta))xy & y \sin(\theta) + (1 - \cos(\theta))xz \\ z \sin(\theta) + (1 - \cos(\theta))xy & 1 + (1 - \cos(\theta))(y^2 - 1) & -x \sin(\theta) + (1 - \cos(\theta))yz \\ -y \sin(\theta) + (1 - \cos(\theta))xz & x \sin(\theta) + (1 - \cos(\theta))yz & 1 + (1 - \cos(\theta))(z^2 - 1) \end{bmatrix} \quad (4.73)$$

This rotation matrix is used to rotate the direction of the positron through the angle Θ around an (arbitrary) axis perpendicular to the original direction as determined by the kinematics. There is, however another angle that is not determined by the kinematics because of azimuthal symmetry, and that is a rotation around the initial momentum. For this, a random angle (Φ_r) is generated and the final momentum is rotated around the initial momentum by this angle, using the same rotation matrix.

This simulation is continued until the positron either escapes the detector (a very rare event) or its energy falls below the cutoff energy. This simulation does not contain in-flight annihilation of the positron since we are interested in the energy of the positron deposited in the detector and if it were annihilated in flight, it would deposit less than its total energy in the detector. However, it can be added fairly simply.

Chapter 5 - RESULTS

The biggest concern and reason for this simulation is to find out if the positron can be used to find the origin of the reaction, (position where the antineutrino interacts with the proton) and if this will still allow the necessary direction sensitivity. Since there are many ways to get information from a simulation, the one which would be closest to the output from real detectors was used. Since the positron loses energy almost continuously (from a measurement point of view) as it propagates through the material, and the duration of its propagation is so small (of the order of nanoseconds), it was decided that the most sensible output would be to calculate the average position at which the energy is deposited. This should reflect what can be determined from a system of photo-multiplier tubes (PMT). Of course, the resultant distances from the origin would be smaller than the actual maximum range of the positron. This average position is then used as the position given by the positron or also referred to as the position of the positron.

After calculating the position of the positron and of the neutron, a line can be drawn between them and that line will be in the direction of the antineutrino before the interaction. Of course the nature of the neutron's propagation in the medium means that the direction given by one event can deviate from the real direction by anything up to 180° (i.e. completely wrong). Therefore, as with all neutron experiments, a lot of statistics are required to determine the general direction that the antineutrinos are moving in. This is one of the factors investigated here.

There are two main spectra of antineutrinos that are important to this investigation. They are the spectrum of natural antineutrinos from natural sources of uranium and thorium (Fig. 5.1) and the spectrum of antineutrinos from nuclear fission reactors (Fig. 5.2) [22]. Note that the horizontal scales are different. The range of energies for the natural spectrum is from 1.85 MeV to 3.5 MeV whereas the range of energies for the reactor spectrum is from 2 MeV to 8.2 MeV. These are the ranges used in the code. The reason for the different starting points is that in the reactor spectrum the number of antineutrinos between 1.8 MeV and 2 MeV is negligible whereas it is very significant for the natural spectrum.

The main concern for the EARTH project is with the spectrum from the natural sources, but for reactor monitoring and initial detector testing, the reactor spectrum is important. In this section results for the natural spectrum will be shown and discussed, but for the sake of completeness, all corresponding graphs using the reactor spectrum have been added to section A.

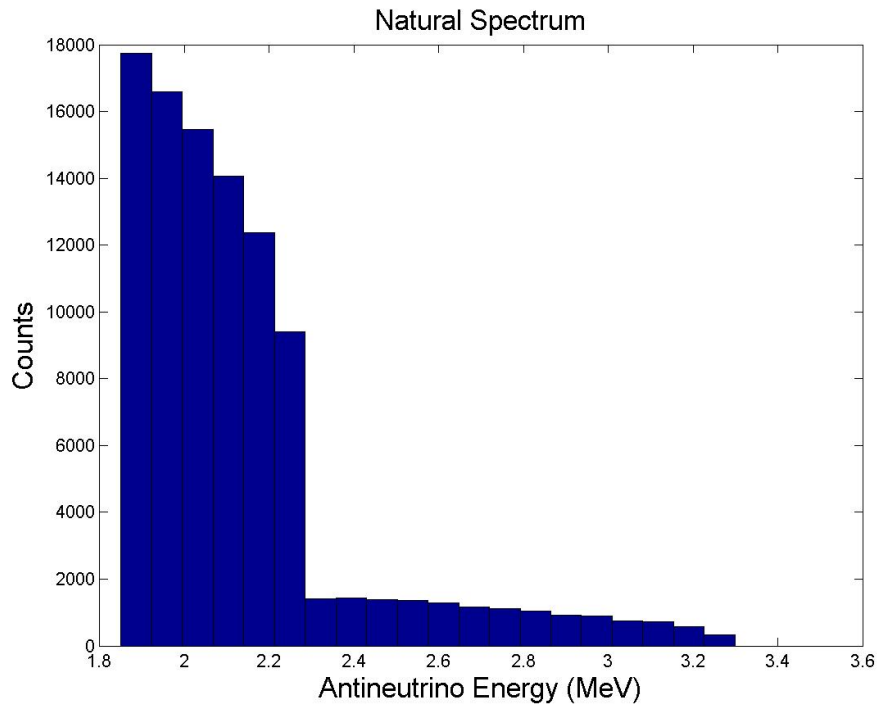


Figure 5.1: Spectrum of antineutrinos coming from natural sources of uranium and thorium [22].

Since the position of the positron (given by the average position where the energy loss is detected) is used as the origin of the reaction, one of the biggest concerns is how large the error is that this assumption causes. Figs. 5.3 and 5.4 show the meanings of Θ_o and Θ_p . If the difference between Θ_o and Θ_p is too large, the deviation of the measured direction from the actual direction of the antineutrino will also be too large.

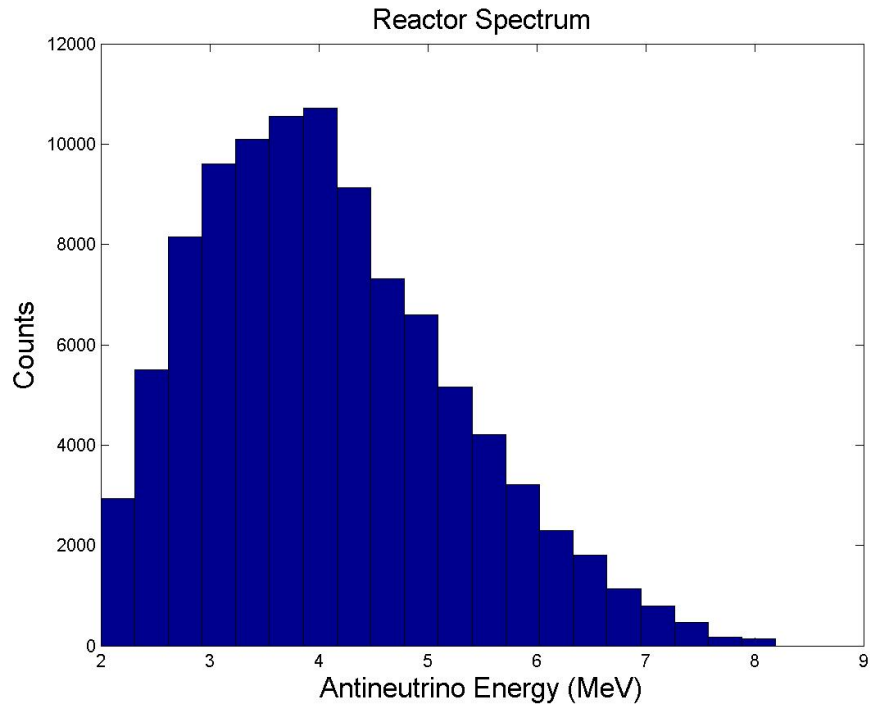


Figure 5.2: Spectrum of antineutrinos coming from nuclear fission reactors [22].

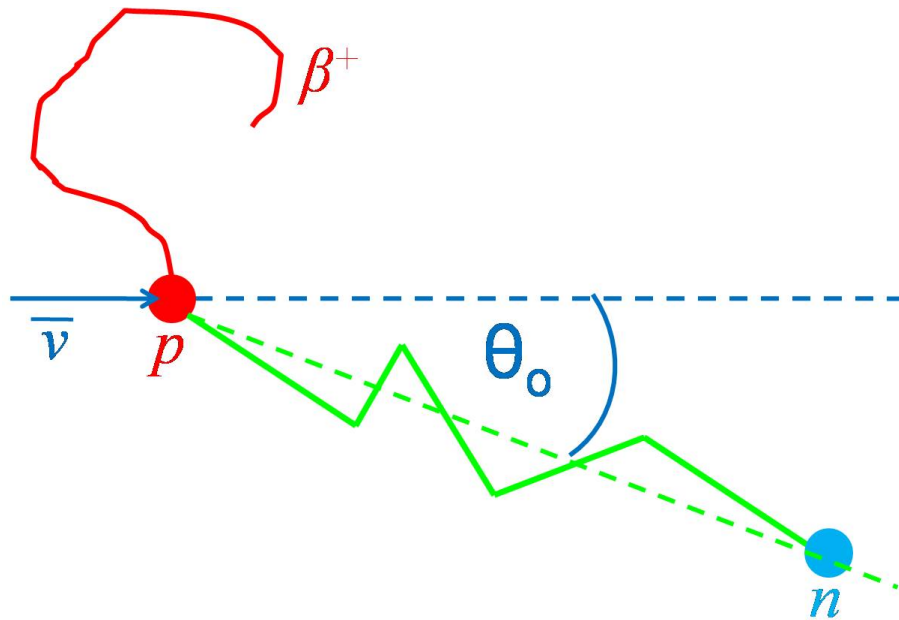


Figure 5.3: Diagram showing the angle θ_o .

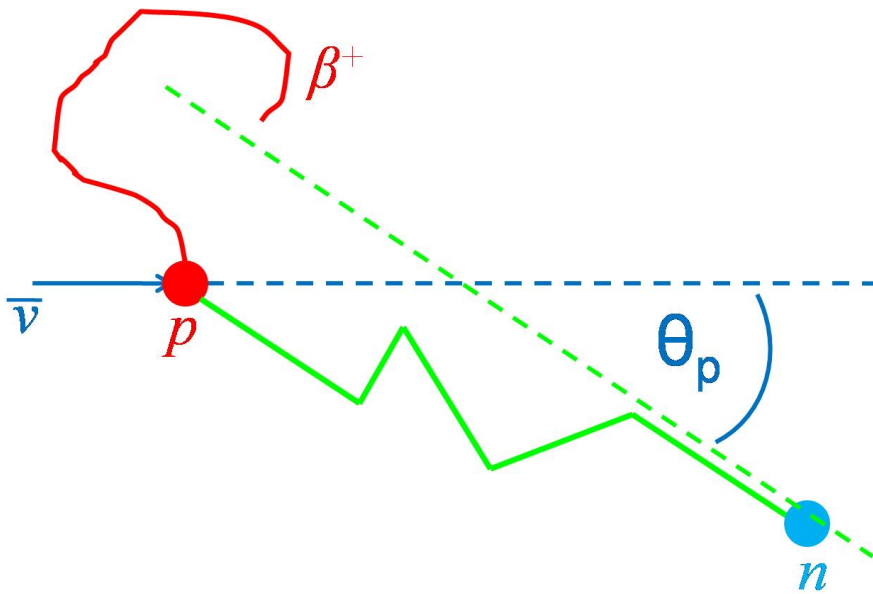


Figure 5.4: Diagram showing the angle θ_p .

5.1 Positron Distribution

The first point to investigate was the distribution of the positron around the origin. The positron transport is essentially a random walk and therefore it is important to know where it is most likely to go, as well as how far from the origin. The distance from the origin is of great concern since if the distance is too big, the difference between Θ_o and Θ_p will be too big and directional sensitivity will be bad.

For Figs. 5.5 and 5.6, a uniform antineutrino energy distribution (i.e. the same amount of antineutrinos for each energy) was used. The positrons seem to range anywhere from 20° to 160° away from the initial direction of the antineutrino, but there is a slow increase toward 90° . When considering the initial reaction when the positron and neutron are formed, it becomes clear that there would be a slightly higher tendency for the positrons to start off going sideways from the initial reaction. What is more important to realise is that the distribution shown in Fig. 5.5 is in fact more isotropic than one would realise at first glance. Consider a sphere with an arbitrary axis through the centre. If one looks at all points at a certain angle from the axis, one would get a circle. For smaller angles (closer to the axis), the circle would be smaller, and have a smaller circumference and contain fewer points. At angles close to 90° the circle would have the full circumference of the sphere. Keeping that picture in mind, if one had a perfectly isotropic distribution, there would be more points around 90° than around 0° . Thus, the distribution in Fig. 5.5 is in fact more isotropic than it seems.

Of course the radial distribution is very important too. It should be stressed that Fig. 5.6 shows the radial energy distribution, and not the radial distribution of the positron. Hence, it does not show the maximum distance the positron will reach from the origin. The positron will generally travel further than that distance. What is important to note though is that there is a definite maximum distance for any specific energy. This is the maximum distance at which the positron will be detected. The other interesting point to make here is that, especially at higher energies, the distribution seems to be spread almost evenly over a distance. For example, at 9 MeV, the highest energy, there are as many counts at 3 cm as there are at 7 cm, and everywhere in-between. There is then a very slight increase toward 9 cm. What this indicates is that even at high energies, we can get a lot of positrons that never stray far from the origin and will therefore still deliver good results. The energies of greatest importance are around 3 and 4 MeV. At those energies, the maximum distance of the energy distribution is around 5 mm. Comparing this to the typical traveling distance of the neutron which is a few centimeters, reveals that the positron distance is at least one order of magnitude smaller. This is already a good sign for the accuracy of using the positron to indicate the origin.

Figure 5.7 comes from [23] and shows the radial distribution of positrons in water. The main differences between the circumstances behind Fig. 5.6 and Fig. 5.7, is that our results were obtained using a combination of H, C and B, whereas their results were obtained in water, as well as the fact that their data shows the distance at which the positron is annihilated, whereas, the results

of the simulation shown here, shows the average position where the energy is deposited. Another important point to notice when comparing the two figures is that the energy in Fig. 5.6 is the energy of the antineutrino and not the energy of the positron, which means that the 1.190 MeV in Fig. 5.7 compares to roughly 3 MeV in Fig. 5.6. Keeping these differences in mind, this comparison instills more confidence in the correctness of the simulation.

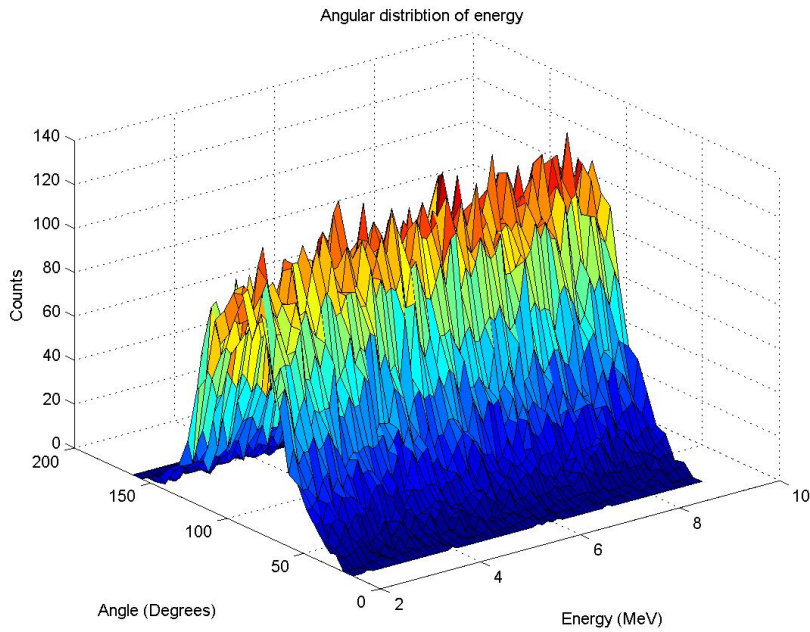


Figure 5.5: Graph of the angular distribution (with 0° being the direction of the antineutrino) of positrons for different energies ranging from 2 MeV to 9 MeV.

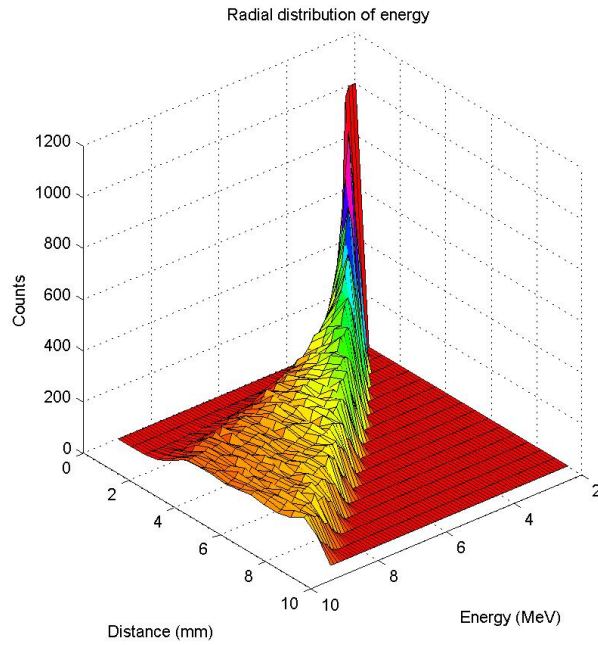


Figure 5.6: Graph of the radial distribution of the positron for different energies ranging from 2 MeV to 9 MeV.

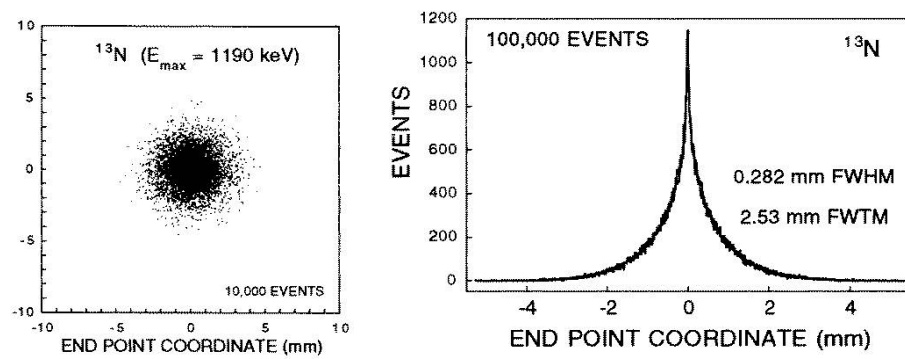


Figure 5.7: Left: calculated distribution of positron annihilation coordinates in water projected onto a plane for ^{13}N source. Right: histogram of x coordinates from positron annihilation point distribution (Fig.7 in [23]).

5.2 Positron And Neutron Together

In this section results will be shown where both the positron and the neutron are simulated. Fig. 5.8 shows the difference between Θ_o and Θ_p . It might appear to be data points spread randomly on a graph, and looking at it naively one might think that the error starts small at around 500 events (the first data point) and that it increases gradually from there, but that is mere chance. There are so many random components to this reaction, that a correlation like that is not impossible, but probably not repeatable. The important deduction that can be made from Fig. 5.8 is that there is no real trend for the difference, but that it doesn't go higher than 0.05° . This is a very good result. Toward the right hand side of the graph the difference does seem to decrease. In contrast to the increase at the beginning, this could be something real, but more data would be required to verify that. Since this is for anti-neutrino detection however, 50 000 events are already much more than can be expected in a reasonable amount of time from real detectors. Furthermore, the error of 0.05° is a very acceptable error.

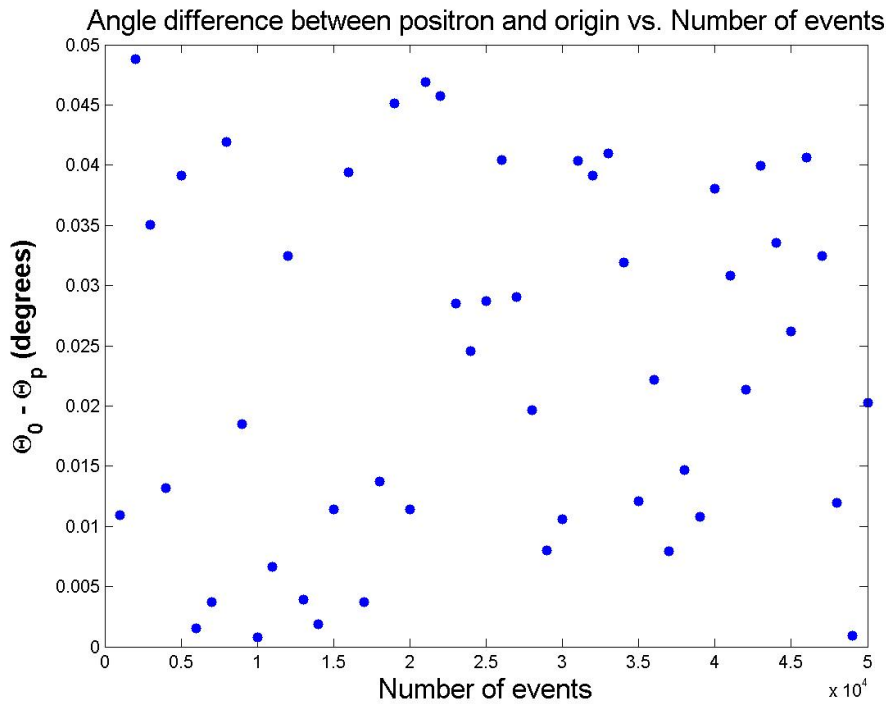


Figure 5.8: Graph of the difference between Θ_o and Θ_p as a function of number of events. (Natural Spectrum)

Figs. 5.9 and 5.10 show angles calculated from single events. Both of them show a rather large deviation, with the peak around 35° . There is only a small difference between the two histograms, confirming the fact that using the positron as the origin is a good assumption. Notice that there are two

small peaks at the top of the histograms. These have not been investigated further, but it may have something to do with the spectrum used. The same phenomenon is visible for the reactor spectrum, but it is less clear.

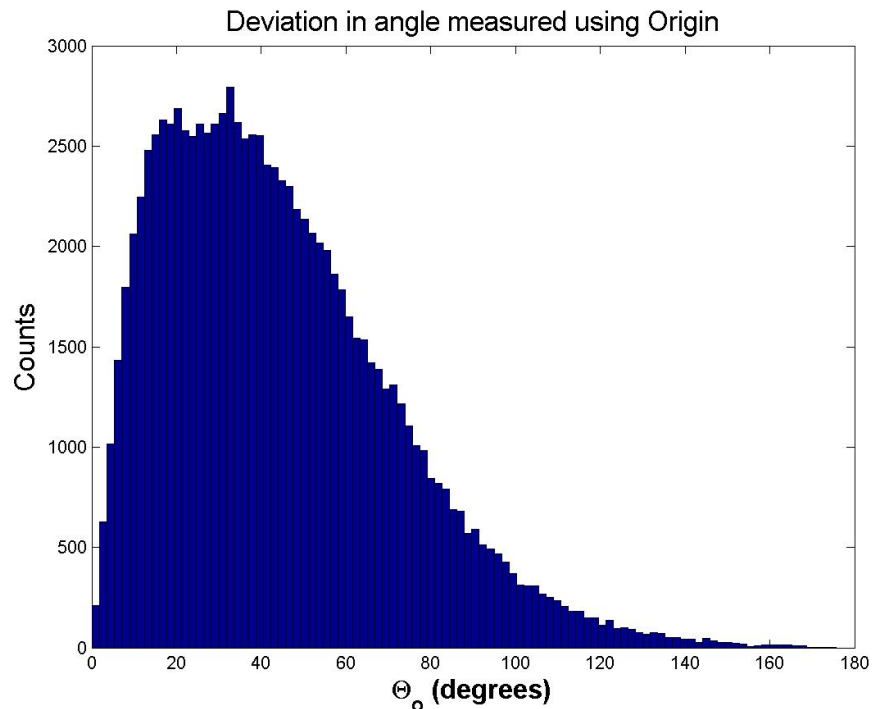


Figure 5.9: Histogram of Θ_o calculated from single events. (Natural Spectrum)

The difference between Θ_o and Θ_p for each event was also investigated. That is to say, for each event, both Θ_o and Θ_p were calculated and the difference was then found. This was plotted in a histogram in Fig. 5.11. Notice that the vertical scale is logarithmic and more than half the events (the data point in the upper left hand side) had a difference of 2° or less. This is the final confirmation that the error caused by using the positron as the origin is very small indeed.

Since it was found that the error caused by using the positron is very small, the origin will be ignored from here on and the positron will be accepted as being the origin.

The next very important question that needs to be answered is: "How many events are necessary in order to get good direction sensitivity?" To answer this question, Fig. 5.12 is considered. It can be seen from the graph (Fig. 5.12) that to get within 2° approximately 3000 events are required. Again, due to the random nature of the experiments, there are a lot of data points lower than that, but the trend of the highest points give the only real information.

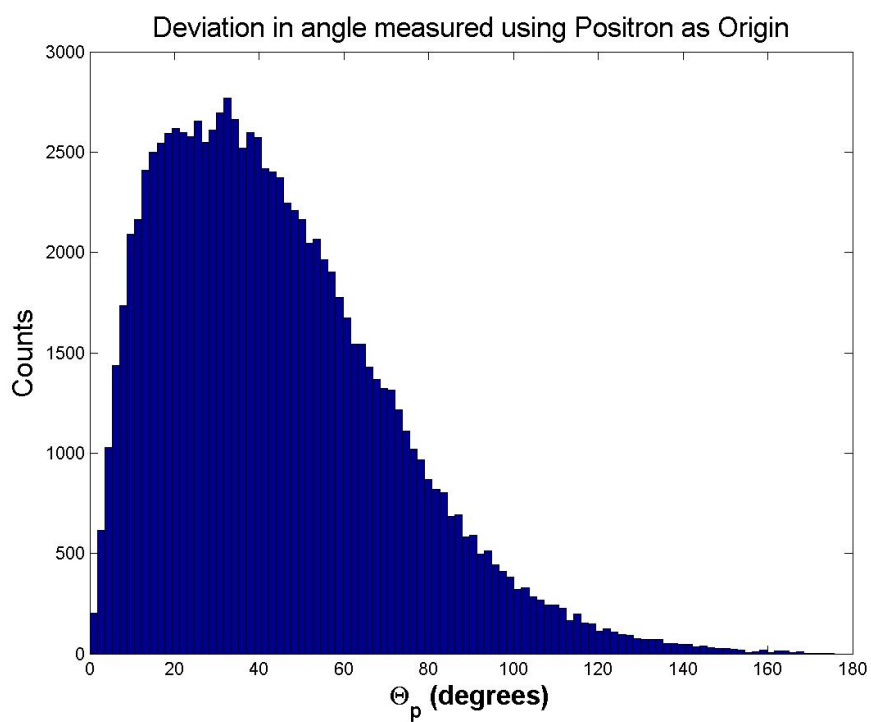


Figure 5.10: Histogram of Θ_p calculated from single events. (Natural Spectrum)

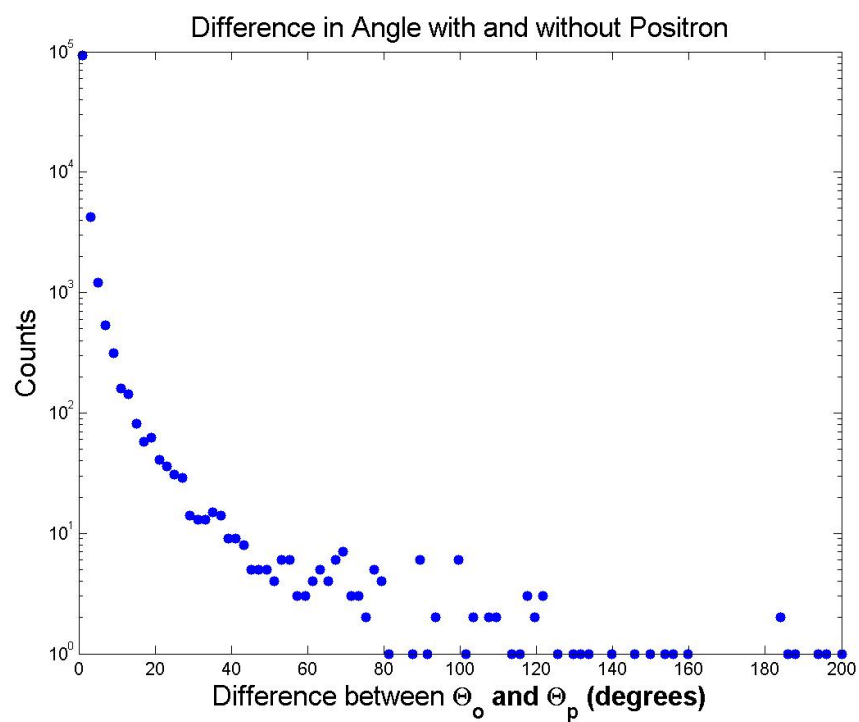
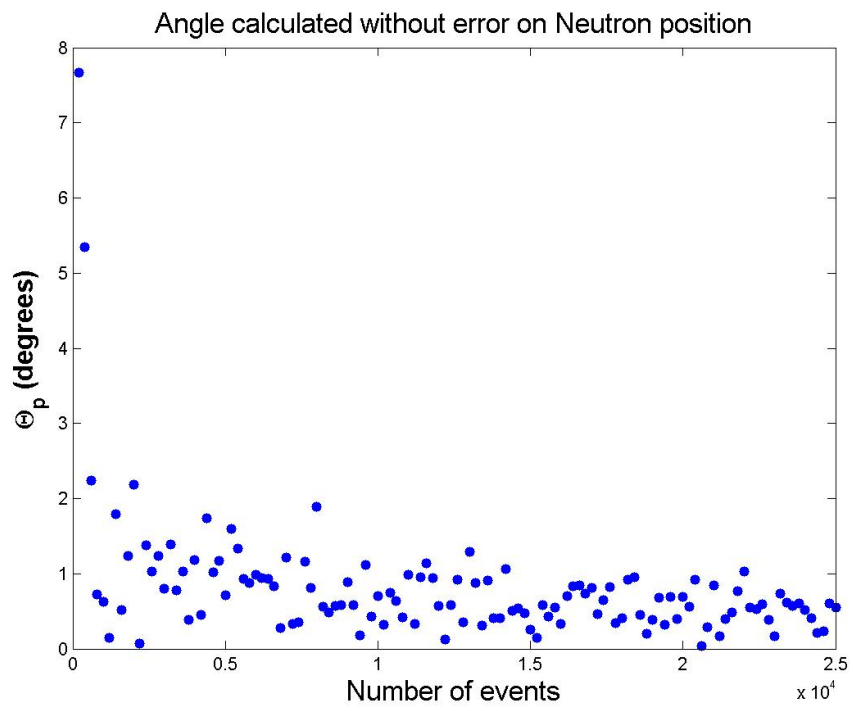


Figure 5.11: Histogram of the difference between Θ_o and Θ_p , calculated from single events. (Natural Spectrum)

Figure 5.12: Θ_p vs. Number of events. (Natural Spectrum)

5.3 Limited Neutron Position Resolution

Of course all these data are obtained from a computer simulation, which means that the final position of the neutron is given to a very high accuracy. In reality though, the limited resolution of the detectors will cause errors in the position of the neutron. In order to investigate this, an error was added to each neutron. Basically a cube with sides of 2 cm was drawn with the neutron's exact position in the centre. A random point inside this cube was then generated and taken as the detected position of the neutron. At the time the data was analyzed, the best resolution available was around 1cm. Hence it was decided to investigate it with worse resolution. Fig. 5.13 shows the same type of graph as Fig. 5.12 but with the error. Notice in the upper title of the graph, that it says 20 mm. This just means that it is a 3 dimensional error and that the sides of the cube are 20 mm long.

Even with this limited resolution on the neutron position, Θ_p should still be smaller than 2° for 3000 events.

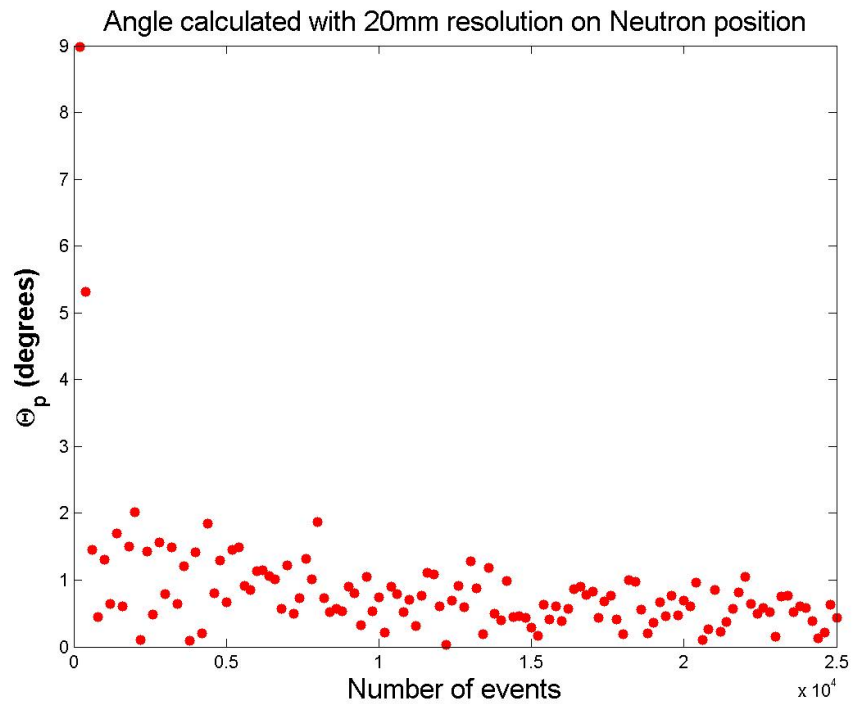


Figure 5.13: Θ_p vs. Number of events with error on the neutron position. (Natural Spectrum)

Chapter 6 - CONCLUSION AND FUTURE PLANS

Through detection of the energy and direction of antineutrinos originating from the earth, it may be possible to obtain valuable information concerning the structure and composition of the planet. With such directional information about these antineutrinos, information can be gained about how radioactive materials are distributed in the Earth and possibly even determine if natural nuclear fission reactors are present. Simulations were done to determine if directional sensitive antineutrino detection is possible by means of the inverse beta decay reaction.

6.1 Error Due To Positron

By drawing a line between the position where the antineutrino is captured and the position where the neutron is captured, a rough idea of the original direction of the antineutrino can be obtained. The exact point of the antineutrino capture can however, not be detected. Using the positron's 'continuous' loss of energy to the scintillator as it propagates, gives a good approximation to the position of the original proton, or the origin of the reaction. Table 6.1 shows the expected difference in direction between using the positron and the actual origin in dependence of number of observed events. The differences for the reactor spectrum are significantly higher since the energies of the antineutrinos are higher and hence the positron travels a greater distance. These differences are however, still very small for a reasonable number of events.

	Natural	Reactor
Events	Difference (°)	Difference (°)
2500	0.052	0.42
25000	0.042	0.42
50000	0.038	0.38

Table 6.1: Table showing the expected difference between using the positron and the actual origin of the reaction for the natural spectrum and the reactor spectrum.

6.2 Required Statistics

For a single event, the deviation from the calculated measured direction to the actual direction of the antineutrino, is of the order of 30° , but this is not surprising as with all neutron detection experiments, one needs a lot of statistics. Table 6.2 shows the expected deviation of our measured direction from the actual direction of the antineutrino. For both spectra we can get quite close to the actual direction with around 2500 events. Beyond this point the increase in accuracy is slow and does not really justify the time it would take to collect more data. This shows us that, even with a limited resolution on the neutron's position, we can get good direction sensitivity with a reasonable number of events.

	Natural	Reactor
Events	Error ($^\circ$)	Error ($^\circ$)
200	4	9.4
1000	3.5	6
2500	2	2.5
10000	1.5	2.0
15000	1.5	2.0
25000	1.0	1.5

Table 6.2: Table showing the expected deviation of measured direction from actual antineutrino direction for the natural spectrum and the reactor spectrum.

The simulations we have done show that there should be no physical reason why directionally sensitive antineutrino detection should not be possible. Advances in detection technology in the next few years should enable us to do these measurements with good accuracy.

6.3 Future Plans

Although all of the subroutines of our simulation program have been tested thoroughly with other widely used programs, another program to test our program as a whole has not yet been found. There are general purpose simulation programs, that can be used to test it, but learning to use them and then test our program would take up more time than is available. Also, it would be better if someone unfamiliar with our program were to test it, since admittedly, if there are errors of thought in the way the simulation was approached, those same errors would be carried over into another simulation program. Nonetheless, the tests done on the subroutines give us great confidence in this simulation program.

A further problem that would require some attention is that of determining the positions of multiple sources of antineutrinos using multiple detectors at different locations around the sources. This problem has great difficulty due to the extreme deviations (some as large as 180°) of some events from the actual

direction, leading to the problem of differentiating between antineutrinos from one point of origin or another.

6.3.1 Current Status Of EARTH Project

The next milestone that the EARTH project is aiming for along the way to making the tomography of radioactive materials inside the Earth, is to build a small detector to place near the core of one of the nuclear reactors at Koeberg Power Station in South Africa. This detector will go by the name of GiZA (Geoneutrinos in South Africa). The design of GiZA has already been finalised and calculations have been done to determine the light output. Fig. 6.1 shows a simple representation of the design of the GiZA detector. It is shaped like a tetrahedron with the PMT's on the four corners. The volume of the target material will be approximately 80 l. On Fig. 6.1 one can see three side panels coming together at the one PMT. These are plastic scintillators that will be used as cosmic vetos. This is not necessarily the final shape of the detectors the project EARTH will use for geoneutrinos, but the shape was chosen to give a good volume of scintillator material and close range to the PMT's. The current shape of GiZA has been determined for a few years and was therefore not influenced by the work done in this thesis. The code that was written does allow for any dimensions of both cylindrical or tetrahedral detectors to be tested though. Therefore, even though there is no confirmation whether or not these calculations will affect the future designs, the facility is in place to test different designs by giving a few bits of input to the program. Various scintillators are presently being tested to find one with all the properties required for direction sensitive antineutrino detection. As soon as a scintillator material is chosen, the detector will be built and measurements of antineutrinos will begin [24].

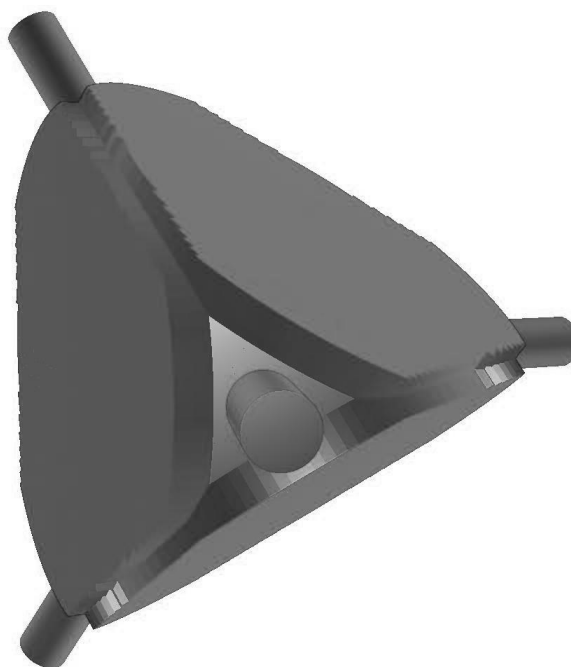


Figure 6.1: Simple model picture of what the GiZA detector will look like.

Appendix A - REACTOR SPECTRUM GRAPHS

In section (5), certain graphs are shown where the natural spectrum was used. In this section the equivalent graphs obtained using the reactor spectrum are shown.

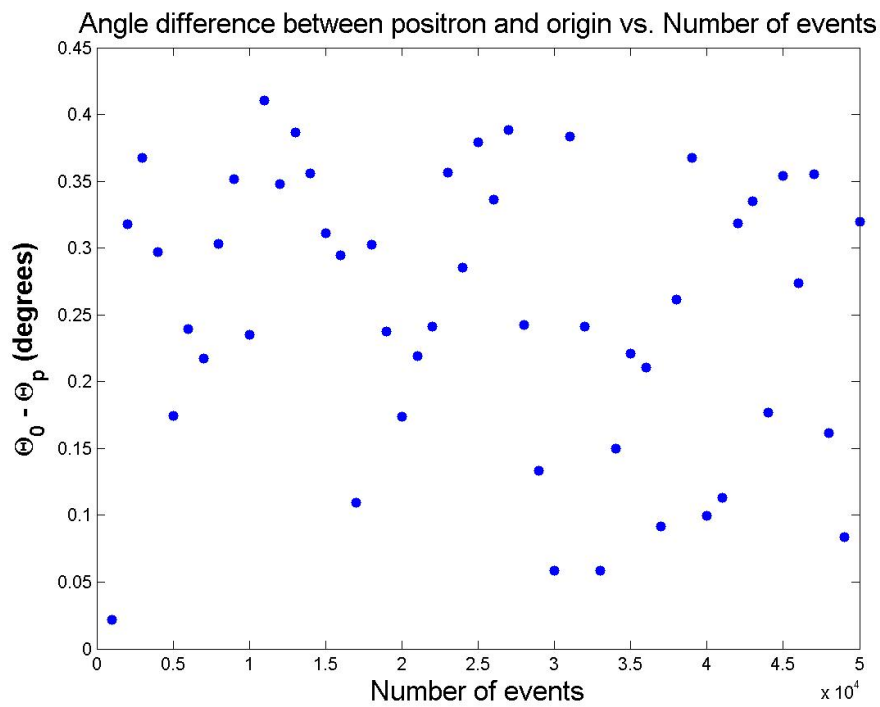


Figure A.1: Graph of the difference between Θ_o and Θ_p as a function of number of events. (Reactor Spectrum)

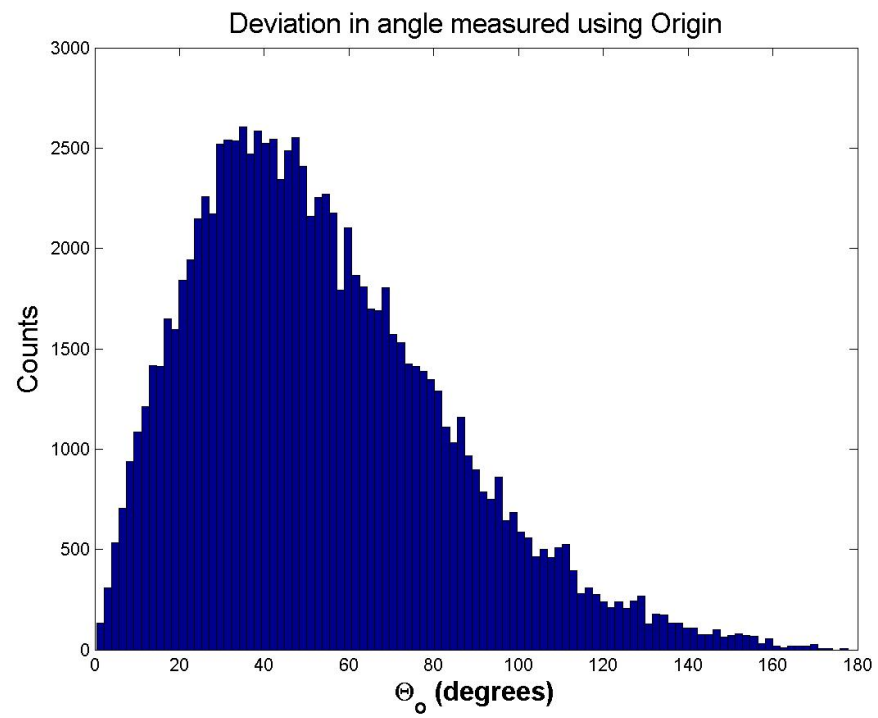


Figure A.2: Histogram of Θ_o calculated from single events. (Reactor Spectrum)

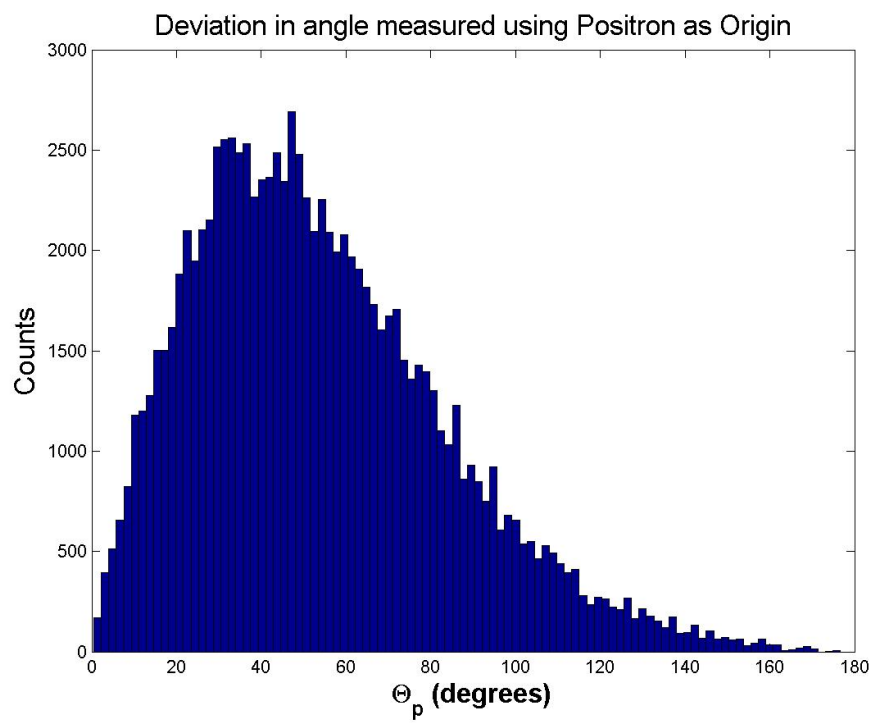


Figure A.3: Histogram of Θ_p calculated from single events. (Reactor Spectrum)

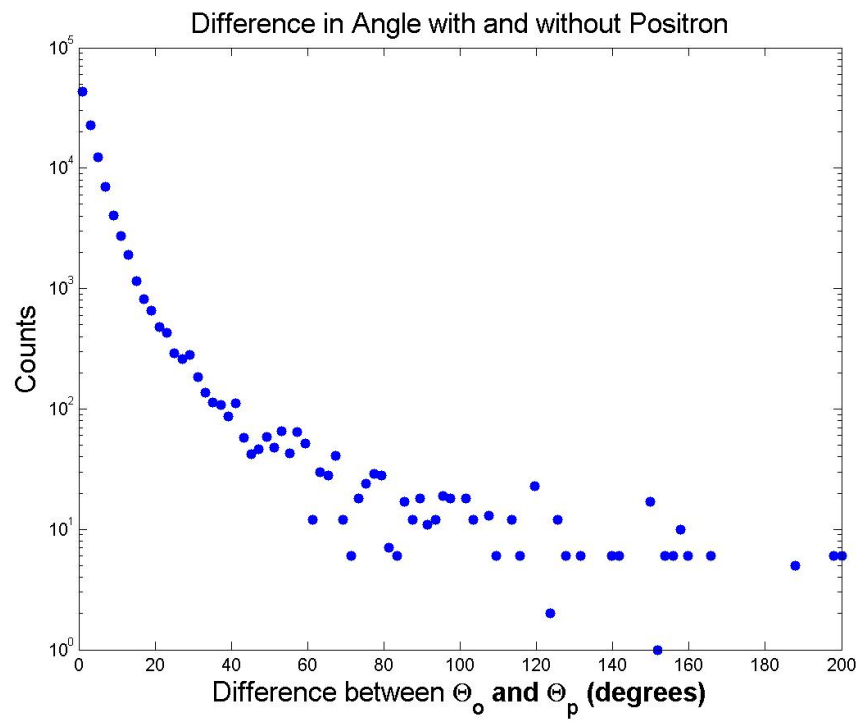


Figure A.4: Histogram of the difference between Θ_o and Θ_p , calculated from single events. (Reactor Spectrum)

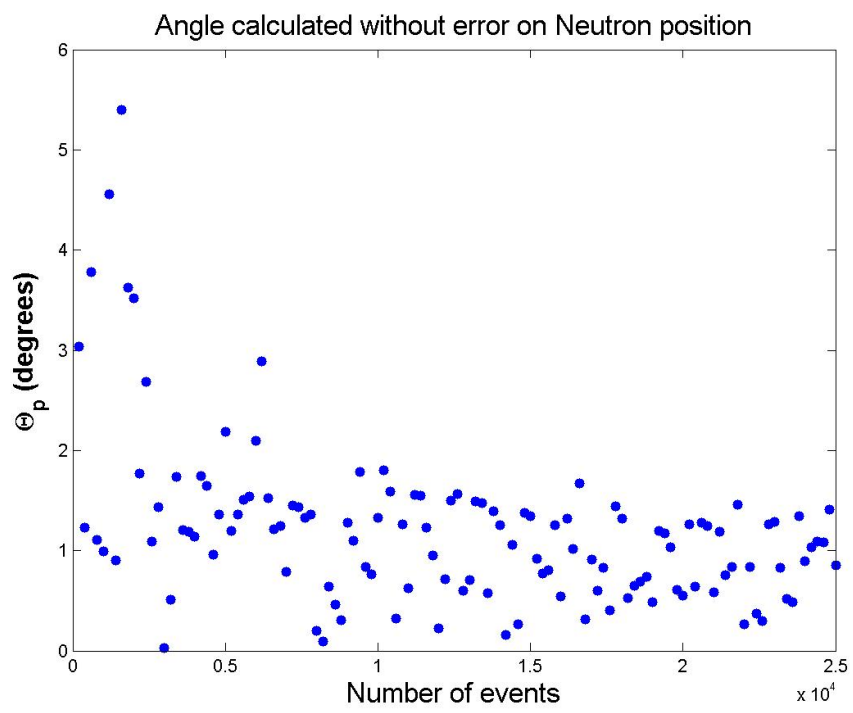


Figure A.5: Θ_p vs. Number of events. (Reactor Spectrum)

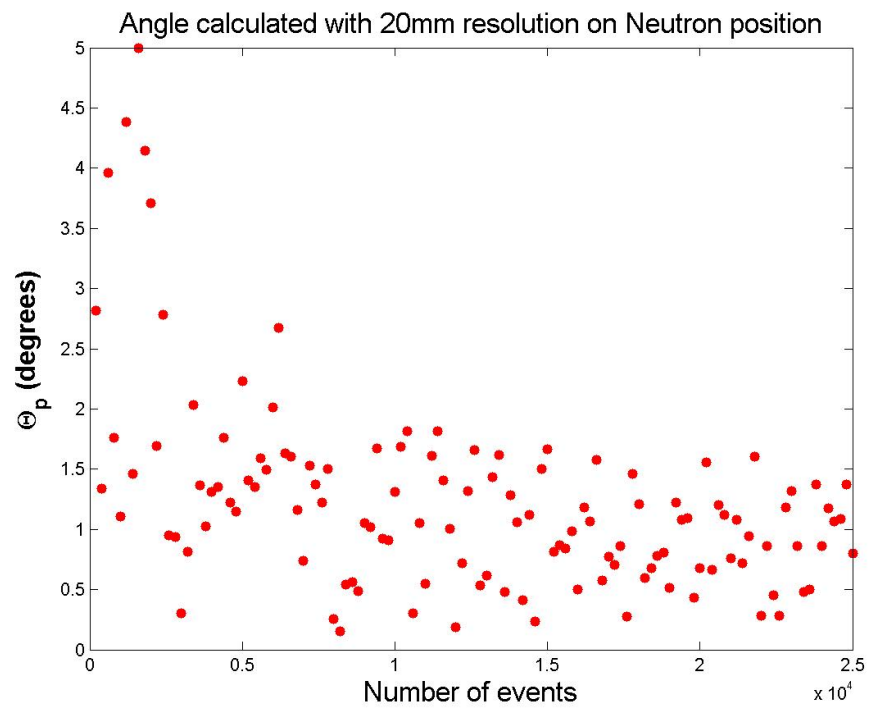


Figure A.6: Θ_p vs. Number of events with error on the neutron position. (Reactor Spectrum)

Appendix B - SIMULATION CODE

This section contains all the code for the simulation. Some sections are commented out (indicated by '//') for one of two reasons. Either it is an actual comment, or it is a part that was not necessary for the final results. An example of this is giving each and every position the positron crosses as output. This would slow the program down and give tremendous amounts of unnecessary output. Other examples are different forms of output being required. Therefore, 'uncommenting' some of the sections might make the program not work or not compile. The version that is given here is in the form that was used to obtain the final results.

In some instances the lines of code are too long for the page width. In these cases, the line of code continues on the next line, preceded by -->. This is not true C++ code so the program will not work with those parts in.

B.1 Main

```
#include <iostream.h>
#include <fstream.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <iomanip.h>
#include "include\MathFnc.cpp"
#include "include\InitializationFnc.cpp"
#include "include\WallsFnc.cpp"
#include "include\PositronScatteringFnc.cpp"
#include "include\NeutronFnc.cpp"

int main()
{
shape = 2; // '0' for Cylinder, '1' for tetrahedron, '2' for none
-->and (0,0,0) origin
int NEvents, Energies = 1,type; //0:Test 1:238 U 2:232 Th
-->3:Th + U 4: Reactor

type = 0;
```



```

srand(time(0));
m_e = 0.510998918; // mass of electron in MeV
m_n = 939.565360;
m_p = 938.272029;
pi = 3.14159265;
r_e = 2.81179402894; // classical radius of electron in fm
c = 2.998e+23; // c in fm
Percentage_B = 5;

double *pV_p, *pV_n, *p4mom_p, *pR_p, *pR_n, E_min = 0, E_max = 0,
--> *pObservedRelativePosition, ObservedAngle,*pZaxis,Origin[3],
-->*pNeutronRelativePosition,NeutronAngle,t_n;

while ((type != 1) && (type != 2) && (type != 3) && (type != 4) &&
-->(type != 5) && (type != 6))
{
cout << "\nType of Spectrum?\n\n1: 238U \n2: 232Th \n3: Th + U (Natural)
-->\n4: Reactor \n5: Uniform Energy Distribution \n6: Single Energy\n";
cin >> type;
}
if (type == 5)
{
cout<<"Minimum Energy (MeV)\n";
cin>>E_min;
while (E_min <= 1.8)
{
cout<<"Energy must be higher than 1.8 MeV\n";
cin>>E_min;
}
cout<<"Maximum Energy (MeV)\n";
cin>>E_max;
while (E_max <= 1.8)
{
cout<<"Energy must be higher than 1.8 MeV\n";
cin>>E_max;
}
}
if (type == 6)
{
cout<<"Energy (MeV)\n";
cin>>E_max;
while (E_max <= 1.8)
{
cout<<"Energy must be higher than 1.8 MeV\n";
cin>>E_max;
}
}
}

```

B.1. MAIN

55

```

cout<<"Enter number of events\n";
cin >>NEvents;

int tUpdate = 100;

cout<<"Press 1 for advanced options or any other key to continue\n";
int AdvancedOptions;
cin>>AdvancedOptions;

if (AdvancedOptions == 1)
{
// cout<<"Information update interval\n";
// cin>>tUpdate;

cout<<"Shape: \n0: Cylinder \n1: Tetrahedron \n2: None (all events start
-->at origin) \n";
cin>>shape;
switch (shape)
{
case 0:
{
cout<<"Radius of Cylinder in mm:\n";
cin>>RadiusCyl;
RadiusCyl = RadiusCyl*1e12;
cout<<"Length of Cylinder in mm:\n";
cin>>LengthCyl;
LengthCyl = LengthCyl*1e12;
break;
}
case 1:
{
cout<<"Length of tetrahedron from corner to corner in mm:\n";
cin>>LengthTet;
LengthTet= LengthTet/sqrt(2)*1e12;
// cout<<"Radius in mm of PMT's a the corners:\n";
// cin>>RadiusPMT;
break;
}

}
}

int Missing = 0,NScats[NEvents],Escapees = 0,Captives = 0,PEscapees = 0;
int *pNeutronData; //[NScats, Captives, Escapees, Missing]
double *pPositronData;
double V_n[4]={0,0,0,0},V_p[4]={0,0,0,0},R_p[3]={0,0,0},E,R_n[3] [NEvents];

```

```

pZaxis = new double[3];
pZaxis[0] = 0;
pZaxis[1] = 0;
pZaxis[2] = 1;
pV_n = new double[4];
pV_p = new double[4];
p4mom_p = new double[4];
pR_p = new double[3];
pR_n = new double[3];
pObservedRelativePosition = new double[3];
pNeutronRelativePosition = new double[3];
pNeutronData = new int[4];
pPositronData = new double[8]; /* Escape (1 => Escaped)
Maximum range
T at maximum range
angle between original and furthest point
Area subtended
Average Angle
Average Distance
Time
*/
for (int l=0;l<4;l++)
{
pNeutronData[l] = 0;
}

pV_n = V_n;
pV_p = V_p;
pR_p = R_p;

int timestamp = time(0);

time_t rawtime, timenow;
time ( &rawtime );
int InitialTime;
InitialTime = rawtime;

double year = rawtime/(3600*24*365);

printf ( "Year: %1d ",(rawtime/(3600*24*(365*4+1)/4)+1970));
printf ( "Day: %1d ",(rawtime/(3600*24))%((365*4+1)/4) );
printf ( "Time: %1d:%1d:%1d\n ",(rawtime/(3600))%(24), (rawtime/60)%60 ,
-->(rawtime)%60 );

char filename3[20];
sprintf(filename3,"Data\\Observed Angle_%1d_%1d_%1d_%1d_%1d_Type_%d.csv",
-->(rawtime/(3600*24*(365*4+1)/4)+1970),((rawtime/(3600*24))%((365*4+1)/4)),

```

```

-->((rawtime/(3600))%(24)),((rawtime/60)%60),((rawtime)%60),type);
ofstream OA(filename3,ios::app);

OA<<"Number of Events: "<<NEvents<<"\nSpectrum type: "<<type<<"\n";
  switch(shape)
  {
  case 0:
  {
  OA<<"Shape: Cylinder\n";
  OA<<"Radius: "<<RadiusCyl*1e-12<<"mm\n";
  OA<<"Length: "<<LengthCyl*1e-12<<"mm\n\n";
  break;
  }
  case 1:
  {
  OA<<"Shape: Tetrahedron\n";
  OA<<"Corner to Corner: "<<sqrt(2)*LengthTet*1e-12<<"mm\n\n";
  break;
  }
  case 2:
  {
  OA<<"Shape: None (events start at origin)\n\n";
  break;
  }
  }

OA<<"Energy,Angle,N_Angle,R_n(mm),R_p(mm),P-N(mm),O_x,O_y,O_z,n_x,n_y,n_z,
-->p_x,p_y,p_z,t_p(?),t_n(?)\n";
OA<<setprecision(4);

for (int l = 0; l < NEvents; l++)
{
if (type != 6)
{
  E = spectrum(type,E_min,E_max);
}
else
{
  E = E_max;
}
InitPositronFnc(E,pV_n,pV_p,p4mom_p);
RandomOrigin(pR_n);

for (int i=0;i<3;i++)
{
pR_p[i] = pR_n[i];
Origin[i] = pR_n[i];
}
for (int i=0;i<7;i++)
{

```

```

pPositronData[i] = 0;
}

PositronScatteringFnc(pR_p,pV_p,p4mom_p,E,l,pPositronData);
//Note: This changes pR_p from where the Positron is, to where we observe it
-->to be.
t_n = NeutronCapture(pV_n,pR_n,pNeutronData);

for(int i = 0;i < 3;i++)
{
pObservedRelativePosition[i] = pR_n[i] - pR_p[i];
pNeutronRelativePosition[i] = pR_n[i] - Origin[i];
}
ObservedAngle = acos( Dot3(pObservedRelativePosition,pZaxis)/
-->(Vec3Mag(pObservedRelativePosition))); //In radians
NeutronAngle = acos( Dot3(pNeutronRelativePosition,pZaxis)/
-->(Vec3Mag(pNeutronRelativePosition)));

OA<<E<<,"<<ObservedAngle*180/pi<<","<<NeutronAngle*180/pi<<","<<
-->Vec3Mag(pNeutronRelativePosition)*1e-12<<","<<sqrt( pow(pR_p[0]-Origin[0],2)
-->+ pow(pR_p[1]-Origin[1],2) + pow(pR_p[2]-Origin[2],2) )*1e-12<<","<<
-->Vec3Mag(pObservedRelativePosition)*1e-12<<","<<Origin[0]*1e-12<<","<<
-->Origin[1]*1e-12<<","<<Origin[2]*1e-12<<","<<pR_n[0]*1e-12<<","<<
-->pR_n[1]*1e-12<<","<<pR_n[2]*1e-12<<","<<pR_p[0]*1e-12<<","<<pR_p[1]*1e-12<<
-->","<<pR_p[2]*1e-12<<","<<pPositronData[7]<<","<<t_n<<"\n";

if (pPositronData[0] == 1)
{
PEscapeses +=1;
}

NScats[1] = pNeutronData[0];

for(int i = 0;i <3 ;i++)
{
R_n[i][1] = pR_n[i];
}

if (l%tUpdate == 1)
{
time (&timenow);
int minutes, seconds;
float time_remaining, EventsLeft;

EventsLeft = NEvents - l;
time_remaining =((EventsLeft/l)*(timenow - rawtime)*1000);
seconds = (int)time_remaining/1000;
printf ("%f MeV\tEvent %d\t%.2d:%.2d:%.2d ellapsed\t %.2d:%.2d:%.2d
-->remaining\n",E,l,((timenow-rawtime)/3600), ((timenow-rawtime)/60)%60,
-->(timenow-rawtime)%60,seconds/3600,(seconds/60)%60, seconds%60);

```

```
cout<<"Rawtime = "<<rawtime<<"\tTimenow = "<<timenow<<"\n";
}
// }
}

double MeanZDisplacement = 0,MeanScatterings = 0;
for(int i = 0;i<NEvents;i++)
{
MeanZDisplacement += R_n[2][i];
MeanScatterings += NScats[i];
}
MeanZDisplacement = MeanZDisplacement/NEvents; //Still in fm.
MeanScatterings = MeanScatterings/NEvents;

double MeanYDisplacement = 0;
for(int i = 0;i<NEvents;i++)
{
MeanYDisplacement += R_n[1][i];
}
MeanYDisplacement = MeanYDisplacement/NEvents;

cout<<"Captives = "<<pNeutronData[1]<<"\tEscapees = "<<pNeutronData[2]<<
-->"\tMissing = "<<pNeutronData[3]<<"\n";
time_t finaltime;
time ( &finaltime );

cout<<"Final time = "<<finaltime<<"\n";

printf ("NEvents: %d\nTotal time: %1d s\n",NEvents,finaltime - rawtime);
}
```

B.2 MathFnc

```
void Vec4Sum2(double* v1,double* v2,double* pSum)
{
for (int i=0;i<4;i++)
{
pSum[i]=v1[i]+v2[i];
}
}

void Vec3Sum2(double* v1,double* v2,double* pSum)
{
for (int i=0;i<3;i++)
{
pSum[i]=v1[i]+v2[i];
}
}

double Vec4Mag(double* vec)
{
return sqrt( pow(vec[0],2) - pow(vec[1],2) - pow(vec[2],2) - pow(vec[3],2));
}

double Vec3Mag(double* vec)
{
return sqrt(vec[0]*vec[0]+vec[1]*vec[1]+vec[2]*vec[2]);
}

double Dot4(double* v1,double* v2)
{
// 4-vector dot product (note negatives)
return (v1[0]*v2[0]-v1[1]*v2[1]-v1[2]*v2[2]-v1[3]*v2[3]);
}

double Dot4Pos(double* v1,double* v2)
{
// 4-vector dot product (note negatives)
return (v1[0]*v2[0]+v1[1]*v2[1]+v1[2]*v2[2]+v1[3]*v2[3]);
}

double Dot3(double* v1,double* v2)
{
return (v1[0]*v2[0]+v1[1]*v2[1]+v1[2]*v2[2]);
}

void CrossProd(double* v1,double* v2,double* prod)
{
prod[0]=v1[1]*v2[2]-v1[2]*v2[1];
prod[1]=v1[2]*v2[0]-v1[0]*v2[2];
```

```

prod[2]=v1[0]*v2[1]-v1[1]*v2[0];
}

// void MatrixMultiplication3x3(double* M1,double* M2,double** product)
//Untested!!!
// //Untested!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
// {
// for (int i=0;i<3;i++) //Rows
// {
// for (int j=0;i<3;i++) //Columns
// {
// product[i][j] = M1[i][0]*M2[j][0] + M1[i][1]*M2[j][1] + M1[i][2]*M2[j][2];
// }
// }
// }

void Boostz(double Beta, double Gamma, double* p4mom)
{
double V0 = Gamma*p4mom[0] + Gamma*Beta*p4mom[3];
double V3 = Gamma*p4mom[3] + Gamma*Beta*p4mom[0];
p4mom[0] = V0;
p4mom[3] = V3;
}

void Boost(double *p4mom, double *pV)
{
// double cc = 2.998e+23;
double Beta[4];
double V[4];
for (int i=0;i<4;i++)
{
Beta[i] = -pV[i];
// cout<<"Beta["<<i<<" = "<<Beta[i]<<"\n";
// cout<<"p4mom["<<i<<" = "<<p4mom[i]<<"\n";
}
// cout<<"Beta[0] = "<<Beta[0]<<"\n";
double Gamma = 1/sqrt(1-pow(Beta[0],2));
// cout <<"Gamma = "<<Gamma<<"\n";

// V[0] = p4mom[0] - pV[1]*p4mom[1] - pV[2]*p4mom[2] - pV[3]*p4mom[3];
// V[1] = -pV[1]*p4mom[0] + p4mom[1];
// V[2] = -pV[2]*p4mom[0] + p4mom[2];
// V[3] = -pV[3]*p4mom[0] + p4mom[3];

// V[0] = Gamma*p4mom[0] + Beta[1]*Gamma*p4mom[1] + Beta[2]*Gamma *
-->p4mom[2] + Beta[3]*Gamma*p4mom[3];
// V[1] = Beta[1]*Gamma*p4mom[0] + (1+(Gamma - 1)*pow(Beta[1],2)/
-->pow(Beta[0],2)) *p4mom[1] + (Gamma - 1)*Beta[1]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] + (Gamma - 1)*Beta[1]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];

```



```

// V[2] = Beta[2]*Gamma*p4mom[0] + (Gamma - 1)*Beta[2]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] + (1+(Gamma - 1)*pow(Beta[2],2)/
-->pow(Beta[0],2))*p4mom[2] + (Gamma - 1)*Beta[2]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];
// V[3] = Beta[3]*Gamma*p4mom[0] + (Gamma - 1)*Beta[3]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] + (Gamma - 1)*Beta[3]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] + (1+(Gamma - 1)*pow(Beta[3],2)/
-->pow(Beta[0],2))*p4mom[3];

// V[0] = Gamma*p4mom[0] + Beta[1]*Gamma*p4mom[1] + Beta[2]*Gamma
-->*p4mom[2] + Beta[3]*Gamma*p4mom[3];
// V[1] = -Beta[1]*Gamma*p4mom[0] - (1+(Gamma - 1)*pow(Beta[1],2)/
-->pow(Beta[0],2)) *p4mom[1] -(Gamma - 1)*Beta[1]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] - (Gamma - 1)*Beta[1]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];
// V[2] = -Beta[2]*Gamma*p4mom[0] - (Gamma - 1)*Beta[2]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] - (1+(Gamma - 1)*pow(Beta[2],2)/
-->pow(Beta[0],2))*p4mom[2] - (Gamma - 1)*Beta[2]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];
// V[3] = -Beta[3]*Gamma*p4mom[0] - (Gamma - 1)*Beta[3]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] -(Gamma - 1)*Beta[3]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] - (1+(Gamma - 1)*pow(Beta[3],2)/
-->pow(Beta[0],2))*p4mom[3];

V[0] = Gamma*p4mom[0] - Beta[1]*Gamma*p4mom[1] -Beta[2]*Gamma
-->*p4mom[2] -Beta[3]*Gamma*p4mom[3];
V[1] = -Beta[1]*Gamma*p4mom[0] + (1+(Gamma - 1)*pow(Beta[1],2)/
-->pow(Beta[0],2)) *p4mom[1] + (Gamma - 1)*Beta[1]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] + (Gamma - 1) *Beta[1]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];
V[2] = -Beta[2]*Gamma*p4mom[0] + (Gamma - 1)*Beta[2]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] + (1+(Gamma - 1)*pow(Beta[2],2)/
-->pow(Beta[0],2))*p4mom[2] + (Gamma - 1)*Beta[2]*Beta[3]/
-->pow(Beta[0],2)*p4mom[3];
V[3] = -Beta[3]*Gamma*p4mom[0] + (Gamma - 1)*Beta[3]*Beta[1]/
-->pow(Beta[0],2) *p4mom[1] + (Gamma - 1)*Beta[3]*Beta[2]/
-->pow(Beta[0],2)*p4mom[2] + (1+(Gamma - 1)*pow(Beta[3],2)/
-->pow(Beta[0],2))*p4mom[3];

for (int i=0;i<4;i++)
{
// cout<<"p4mom["<<i<<" = "<<V[i]<<"\n";
p4mom[i]=V[i];
// cout<<"p4mom["<<i<<" = "<<V[i]<<"\n";
}
cout<<"p4mom magnitude = "<<sqrt(pow(p4mom[1],2)+pow(p4mom[2],2)+
-->pow(p4mom[3],2))<<"\n";
}

```

B.3 InitializationFnc

```

double m_e,m_n,m_p,pi,c,r_e,Percentage_B; //Globals for program
int shape;

void InitPositronFnc(double E,double *pV_n,double *pV_p,double *pp_4)
{
double *pp_1,*pp_2,*pp_3,*pv4_refer1;

double p_1[4]={E,0.0,0.0,E}; //4-momentum of Anti-neutrino
double p_2[4]={m_p,0.0,0.0,0.0}; //4-momentum of Proton
double p_3[4]={0,0,0,0}; //4-momentum of Neutron
double p_4[4]={0,0,0,0}; //4-momentum of Positron
double v4_refer1[4]={0,0,0,0}; //Dummy 4-vector

pp_1=new double [4];
pp_2=new double [4];
pp_3=new double [4];
pv4_refer1=new double [4];

pp_1=p_1;
pp_2=p_2;
pp_3=p_3;
pv4_refer1=v4_refer1;

Vec4Sum2(pp_1,pp_2,pv4_refer1);

double s,E_cm,E_3,a,magnitude_p_3,r1,r2,r3,phi1,z,Rst,gamma, Beta;

s = Dot4(pv4_refer1,pv4_refer1); //Mandelstam s
E_cm = sqrt(s); //Energy of cms system
-->from definition of mandelstam s
E_3 = (s-pow(m_e,2)+pow(m_n,2))/(2*E_cm);
a = (s-pow((m_n+m_e),2))*(s-pow((m_n-m_e),2));
magnitude_p_3 = 0.5*sqrt(s-2*(pow(m_n,2)+pow(m_e,2))+pow((pow(m_n,2)-
--> pow(m_e,2)),2)/(s)); //Finds the magnitude of neutron momentum from kinematics.

// Random velocity for Neutron
r1=(rand()/(RAND_MAX + 1.0));
r2=(rand()/(RAND_MAX + 1.0));
r3=(rand()/(RAND_MAX + 1.0));
int sgn;
if (r3 < 0.5)
{
sgn = 1;
}
else
{
sgn = -1;
}

```

```

}
// Generate a spherically random velocity with magnitude 1
phi1 = 2.0*pi*r1;
z = 2.0*r2-1.0;
Rst = 1*sin(acos(z));
double velocity[3]={sgn*Rst*cos(phi1),Rst*sin(phi1),z };

// Find Momentum 4 vector of Neutron (in cm system)
p_3[0]=E_3;
p_3[1]=velocity[0]*magnitude_p_3;
p_3[2]=velocity[1]*magnitude_p_3;
p_3[3]=velocity[2]*magnitude_p_3;

// cout<<"Neutron P =\t " <<pp_3[0]<<"\t" <<pp_3[1]<<"\t" <<pp_3[2]<<"\t" <<
--> pp_3[3]<<"\n";

// Boost in the Z direction to get into the lab system
gamma = (p_1[0]+Vec4Mag(pp_2))/(E_cm);
Beta = p_1[3]/(p_1[0]+Vec4Mag(pp_2));
Boostz(Beta,gamma,pp_3);
// cout<<"Neutron P =\t " <<pp_3[0]<<"\t" <<pp_3[1]<<"\t" <<pp_3[2]<<"\t" <<
--> pp_3[3]<<"\n";

// Find 4-momnetum of Positron from conservation of momentum
for(int i=0;i<4;i++)
{
p_4[i] = p_1[i]+p_2[i]-p_3[i];
pp_4[i] = p_4[i];
}

// *****
// P_3 and p_4 are now in the lab frame
// *****

double p_perp_n, theta_n, theta_n2, phi_n, p_perp_p, theta_p, theta_p2, phi_p,
--> T_3, T_4,
--> V0n, V0p;

p_perp_p=sqrt(pow(p_4[1],2)+pow(p_4[2],2));
theta_p=atan(p_perp_p/p_4[3]);
theta_p2=acos(p_4[3]/p_4[0]);
phi_p=atan(p_4[2]/p_4[1]);

T_4 = sqrt(pow(p_4[1],2)+pow(p_4[2],2)+pow(p_4[3],2)); //Relativistic T_lab
V0p = sqrt(1 - (1/pow(1+T_4/m_e,2)));

//*****
T_3 = pp_3[0]-m_n;
V0n = sqrt(2*T_3/m_n);

```

B.3. INITIALIZATIONFNC

65

```

double MagP_n = sqrt(pow(pp_3[1],2)+pow(pp_3[2],2)+pow(pp_3[3],2));

pV_n[0]=V0n;
pV_n[1]=V0n*pp_3[1]/MagP_n;
pV_n[2]=V0n*pp_3[2]/MagP_n;
pV_n[3]=V0n*pp_3[3]/MagP_n;

//*****

pV_p[0]=V0p;
pV_p[1]=V0p*cos(theta_p2);
pV_p[2]=V0p*sin(theta_p)*sin(phi_p);
pV_p[3]=V0p*sin(theta_p)*cos(phi_p);

// cout<<"Neutron V  =\t "<<pV_n[0]<<"\t"<<pV_n[1]<<"\t"<<pV_n[2]<<"\t"<<
--> pV_n[3]<<"\n\n";
}

double spectrum(int type)
{
// int type = 0; //0:Test 1:238 U 2:232 Th 3:Th + U 4: Reactor

int length;
double sum = 0,E_max,E_min,E,E_int,r;
double SpecArray[100];

double TestArray[100] = {1,2,3,4,5,6,6.5,7,7.5,8,7.5,7,6.5,6,5,4,3,2,1,0.9,0.7,
--> 0.6,0.5,0.4};
double U238Array[100] = {0.7897,0.7227,0.7017,0.6813,0.6615,0.6236,0.5878,0.5541,
--> 0.538,0.0971,0.0943,0.0943,0.0915,0.0889,0.0863,0.0838,0.079,0.0767,0.0702,
--> 0.0702,0.0642,0.0624,0.0588 ,0.0554,0.0522,0.0492,0.0451,0.0412,0.0367,0.01};
double Th232Array[100] = {0.4248,0.4005,0.3888,0.3665,0.3455,0.2894,0.2572,0.2354
--> ,0.01};
double UPlusThArray[100] = {0.7897+0.4248,0.7227+0.4005,0.7017+0.3888,
--> 0.6813+0.3665,0.6615+0.3455,0.6236+0.2894,0.5878+0.2572,0.5541+0.2354,
--> 0.538+0.01,0.0971,0.0943,0.0943,0.0915,0.0889,0.0863,0.0838,0.079,
--> 0.0767,0.0702,0.0702,0.0642,0.0624,0.0588,0.0554,0.0522,0.0492,0.0451
--> ,0.0412,0.0367,0.01};
double ReactorSpec[100] = {46,97,136,171,196,217,233,239,241,240,229,213,196,174,
--> 156,139,124,107,92,79,67,57,47,38,30,23,17,12,8,6,5,3};

// switch(type)
// {
// case 0:
// {
// length = 24;
// double SpecArray1[] = {1,2,3,4,5,6,6.5,7,7.5,8,7.5,7,6.5,6,5,4,3,2,1,
--> 0.9,0.7,0.6,0.5,0.4};
// E_min = 2;
// E_max = 5;

```

```

// E_int = (E_max - E_min)/length;
// break;
// }
//
// case 1:
// {
// length = 30;
// double SpecArray1[] = {0.7897,0.7227,0.7017,0.6813,0.6615,0.6236,
--> 0.5878,0.5541,0.538,0.0971,0.0943,0.0943,0.0915,0.0889,0.0863,0.0838,
--> 0.079,0.0767,0.0702,0.0702,0.0642,0.0624,0.0588,0.0554,0.0522,0.0492,
--> 0.0451,0.0412,0.0367,0.01};
// E_min = 1.85;
// E_max = 3.3;
// E_int = (E_max - E_min)/length;
// break;
// }
//
// case 2:
// {
// length = 9;
// double SpecArray1[] = {0.4248,0.4005,0.3888,0.3665,0.3455,0.2894,
--> 0.2572,0.2354,0.01};
// E_min = 1.85;
// E_max = 2.25;
// E_int = (E_max - E_min)/length;
// break;
// }
//
// case 3:
// {
// length = 30;
// double SpecArray1[] = {0.7897+0.4248,0.7227+0.4005,0.7017+0.3888,
--> 0.6813+0.3665,0.6615+0.3455,0.6236+0.2894,0.5878+0.2572,0.5541+0.2354,
--> 0.538+0.01,0.0971,0.0943,0.0943,0.0915,0.0889,0.0863,0.0838,0.079,0.0767,
--> 0.0702,0.0702,0.0642,0.0624,0.0588,0.0554,0.0522,0.0492,0.0451,0.0412,
--> 0.0367,0.01};
// E_min = 1.85;
// E_max = 3.3;
// E_int = (E_max - E_min)/length;
// break;
// }
// }
// double SpecArray = SpecArray1;

switch(type)
{
case 0:
{
length = 24;
SpecArray = TestArray;

```

B.3. INITIALIZATIONFNC

67

```
E_min = 2;
E_max = 5;
E_int = (E_max - E_min)/length;
break;
}

case 1:
{
length = 30;
SpecArray = U238Array;
E_min = 1.85;
E_max = 3.3;
E_int = (E_max - E_min)/length;
break;
}

case 2:
{
length = 9;
SpecArray = Th232Array;
E_min = 1.85;
E_max = 2.25;
E_int = (E_max - E_min)/length;
// double XX = 4;
break;
}

case 3:
{
length = 30;
SpecArray = UPlusThArray;
E_min = 1.85;
E_max = 3.3;
E_int = (E_max - E_min)/length;
// double XX = 6;
break;
}

case 4:
{
length = 32;
SpecArray = ReactorSpec;
E_min = 2;
E_max = 8.2;
E_int = (E_max - E_min)/length;
break;
}
}
//
double CollectiveNormSpectrum[length];
```

```

for (int i = 0;i<length;i++)
{
sum += SpecArray[i];
}

for(int i = 0;i<length;i++)
{
SpecArray[i] = SpecArray[i]/sum;
}
CollectiveNormSpectrum[0] = SpecArray[0];

for (int i = 1;i<length;i++)
{
CollectiveNormSpectrum[i]= CollectiveNormSpectrum[i-1] + SpecArray[i];
}
// cout<<"Final value = "<<CollectiveNormSpectrum[length-1]<<"\n";
r = (rand()/(RAND_MAX + 1.0));

if(r <= CollectiveNormSpectrum[0])
{
E = E_min + (r/CollectiveNormSpectrum[0])*E_int;
}
else
{
for(int i = 1; i <length; i++)
{
if (r <= CollectiveNormSpectrum[i] && r >= CollectiveNormSpectrum[i-1] )
{
E = E_min + i*E_int + ((r-CollectiveNormSpectrum[i-1])/
--> (CollectiveNormSpectrum[i]-CollectiveNormSpectrum[i-1]))*E_int;
}
}
}
if (E <= 1.8)
{
cout<<"E = "<<E<<"\tr = "<<r<<"\n";
}

return E;
}

```

B.4 WallsFnc

```

//*****
// Cylinder (along z(x3)-axis)
//*****

double RadiusCyl = 50 *1e12; //Radius in fm
double LengthCyl = 100 *1e12; //Length in fm

//*****
//*****
// Tetrahedron with corners at (0,0,0), (0,1,1), (1,0,1), (1,1,0)
//*****

double LengthTet = 500 *1e12; //Length of side of cube enclosing tetrahedron in fm
double RadiusPMT; //Radius of Photomultiplier tubes at the corners

//*****

int WallCheck(double *pR,int Escape)
{
while (Escape == 0)
{
if (shape == 0)
{
if (sqrt((pR[0]*pR[0])+(pR[1]*pR[1]))>= RadiusCyl)
{
Escape = 1;
}

if ((pR[2] >= LengthCyl/2) || (pR[2]<= -LengthCyl/2))
{
Escape = 1;
}
}

if (shape == 1) //Tetrahedron check
{
if (pR[2] < pR[0] - pR[1]) //Face 134
{
Escape = 1;
}
if (pR[2] > pR[0] + pR[1]) //Face 123
{
Escape = 1;
}
if (pR[2] < - pR[0] + pR[1]) //Face 124
{

```



```
Escape = 1;
}
if (pR[2] > - pR[0] - pR[1] + 2*LengthTet) //Face 324
{
Escape = 1;
}
}

if (shape == 2)
{
Escape = 0;
}

break;
}
return Escape;
}

void RandomOrigin(double *pR)
{
if (shape == 0)
{
double theta,r,z;
theta = (rand()/(RAND_MAX + 1.0)) * 2*pi; // \
r = sqrt(rand()/(RAND_MAX + 1.0)) * RadiusCyl;
--> // / Random distridution on a disc
z = ((rand()/(RAND_MAX + 1.0))- 0.5) * LengthCyl;

pR[0] = r*cos(theta);
pR[1] = r*sin(theta);
pR[2] = z;
}
if (shape == 1)
{
pR[0] = (rand()/(RAND_MAX + 1.0))*LengthTet;
pR[1] = (rand()/(RAND_MAX + 1.0))*LengthTet;

double Zmin_1,Zmin_2,Zmax_1,Zmax_2,Zmin,Zmax;

Zmax_1 = pR[0] + pR[1];
Zmax_2 = -pR[0] - pR[1] + 2*LengthTet;

if(Zmax_1 < Zmax_2)
{
Zmax = Zmax_1;
}
else
{
Zmax = Zmax_2;
}
}
```

B.4. WALLSFNC

71

```
Zmin_1 = pR[0] - pR[1];
Zmin_2 = -pR[0] + pR[1];

if(Zmin_1 > Zmin_2)
{
Zmin = Zmin_1;
}
else
{
Zmin = Zmin_2;
}

pR[2] = Zmin + (rand()/(RAND_MAX + 1.0))*(Zmax-Zmin);
}

if (shape == 2)
{
for (int i = 0;i < 3;i++)
{
pR[i] = 0;
}
}
}
```

B.5 PositronScatteringFnc

```

//*****
// Positron Function

/* Outputs
Escape (1 => Escaped)
Maximum range
T at maximum range
angle between original and furthest point
Area subtended

*///*****

double MollerScatteringCrossSectionFnc(double Z,double Beta,double Gamma,
--> double x,double y)
{
return (2*pi*pow(r_e,2)*Z)/(pow(Beta,2)*(Gamma-1)) * ( (pow(Gamma-1,2)/
--> pow(Gamma,2)*(0.5-x)) + 1/x + 1/(1-x) - (2*Gamma-1)/pow(Gamma,2)*
--> log((1-x)/x) );
}

double BhabhaScatteringCrossSectionFnc(double Z,double Beta,double Gamma,double B1,
--> double B2,double B3,double B4,double x)
{
return (2*pi*pow(r_e,2)*Z)/(Gamma-1) * ( 1/pow(Beta,2)*(1/x-1) + B1*log(x) +
--> B2*(1-x) - B3/2*(1-pow(x,2)) + B4/3*(1-pow(x,2)) );
}

double TwoPartScatterFnc(double *pp1,double Q)
{
//Calculate Lab Angle using Cos rule
// *****

double P1Squared,P1PrimeSquared,P2PrimeSquared,E1,E2Prime,E1Prime,m12,m22;
double m_1 = m_e;
double m_2 = m_e;
m12 = pow(m_1,2);
m22 = pow(m_2,2);
E1 = pp1[0];
E2Prime = m_2 + Q; //Checked!!! (E'= m + T_lab)
E1Prime = E1 - Q; //Checked!!!
P2PrimeSquared = pow(Q,2)+2*m_2*Q; //Checked!!! (E^2 = m^2 + p^2)
P1PrimeSquared = pow(E1Prime,2)-m12;
P1Squared = pow(E1,2)-m12;

pp1[0] = E1Prime;
pp1[1] = P1PrimeSquared; //pp1 is no longer a proper 4-momentum

```

```

double CosLabTheta = (P1Squared + P1PrimeSquared - P2PrimeSquared)/
--> (2*sqrt(P1Squared)*sqrt(P1PrimeSquared));
return acos(CosLabTheta); //theta in radians
}

void PositronScatteringFnc(double *pR_p,double *pV_p,double *p4mom_p,
--> double E,int Event ,double *pPositronData)
// The total mean free path will be the inverse of the sum of the inverses
--> of these 2 multiplied by their weighting factors caused by densities.

// 1) Bhabha Scattering with C
// 2) Bhabha Scattering with H
// 3) Bhabha Scattering with B

// We must then add in the continuous energy loss caused by Brehmsstrahlung
{ double t,Tcut,N_av,Rho_Scint,NHratNC,MolarMass_C,MolarMass_B,MolarMass_H,
--> Rho_C,Rho_B,Rho_H, Energy,Origin[3],Rmax = 0,Rnow = 0, RelativeR[3]={0,0,0},
--> R_prev,ThetaMax,RmaxT,e,Positron_Area, Semi_Perimeter;
int Escape = 0, Counter = 0;
double Angle[700000], Distance[700000],Qpoint[700000],
--> *pZaxis,EnergyPosition[4]={0,0,0,0};
pZaxis = new double[3];
pZaxis[0] = 0;
pZaxis[1] = 0;
pZaxis[2] = 1;

t = 0; // Time
Energy = m_e/(1-pow(pV_p[0],2)); // Total energy
Tcut = 0.001; //Tcut = 1keV
N_av = 6.02214169e+23;

Positron_Area = 0;

Rho_Scint = 0.874; //g/cm^3 Obtain this from Scintillator specs
NHratNC = 1.213; //Ratio between number of H and C
MolarMass_C = 12.011; //g
MolarMass_H = 1.0079; //g
Rho_C = MolarMass_C*Rho_Scint/(NHratNC*MolarMass_H + MolarMass_C);
--> //Densities in g/cm^3
Rho_H = Rho_Scint - Rho_C;
Rho_C = Rho_C * 1/(1.783e18); //Convert to MeV/C^2fm^3
Rho_H = Rho_H * 1/(1.783e18); //Convert to MeV/C^2fm^3
Rho_B = Rho_Scint*(Percentage_B/100)* 1/(1.783e18);
MolarMass_C = MolarMass_C * 1/(1.783e-21); //Convert to MeV/c^2
MolarMass_H = MolarMass_H * 1/(1.783e-21); //Convert to MeV/c^2
MolarMass_B = 10.811 * 1/(1.783e-21);

for(int i=0;i<3;i++)

```

```

{
Origin[i] = pR_p[i];
}

char filename1[20];
// sprintf(filename1,"Trajectories\\R_%f_MeV_Event_%d.txt",E,Event);
// ofstream PP(filename1,ios::app);
// ofstream PP("Pos_R.txt",ios::app);
// PP <<"Time \t x\t y\t z\t Q\t \T_lab\n";

double *pPosOri4mom,*pDumVec;
pPosOri4mom = new double[3];
pDumVec = new double[3];

pDumVec[0]=0;
pDumVec[1]=0;
pDumVec[2]=1;

for (int i=1;i<4;i++)
{
pPosOri4mom[i-1] = p4mom_p[i];
}

double ThetaOri = acos( Dot3(pPosOri4mom,pDumVec)/(Vec3Mag(pPosOri4mom)
--> *Vec3Mag(pDumVec)));

// int j = 0,printout = 50000;
double T_4 = p4mom_p[0]-m_e;

double e2pi4epsilon = 1.440036204; // e^2/(4*pi*Epsilon0) in Mev.fm

double v2,Beta,Beta2,Gamma,x,y,B1,B2,B3,B4,sigmaBC,sigmaBH,sigmaBB,n_at_C,
--> n_at_H,n_at_B,mfp,r1,dr,N,Z,DeltaEBB,DeltaEB,Q,theta,n1[3],n2[3],n3[3],
--> n_perp[3],normalizer,phi,nn,P1Mag,n_H,n_C;

n_at_C = 6*N_av*Rho_C/MolarMass_C;
n_at_H = 1*N_av*Rho_H/MolarMass_H;
n_at_B = 5*N_av*Rho_B/MolarMass_B;
n_H = N_av*Rho_H;
n_C = N_av*Rho_C*6/12;
N = n_at_H + n_at_C/6 + n_at_B/5;
Z = 6*NHratNC + 5*(Percentage_B/100);

// ofstream ET("Data\\EvsT.dat",ios::app);
// ET<<"Time (s)\tT_lab (MeV)\tQ \n";
int FirstRun = 1;

while (T_4 > Tcut)
{
R_prev = Rnow;

```

```

v2 = pow(pV_p[1],2)+pow(pV_p[2],2)+pow(pV_p[3],2);
Beta = sqrt(v2);
Beta2 = pow(Beta,2);
Gamma = 1/sqrt(1-Beta2);
x = Tcut/(Energy - m_e); //Tcut/(E-mc^2)
y =(1/(Gamma + 1));
B1 = 2-pow(y,2);
B2 = (1-2*y)*(3+pow(y,2));
B3 = pow((1-2*y),2)+pow((1-2*y),3);
B4 = pow((1-2*y),3);

sigmaBC = BhabhaScatteringCrossSectionFnc(6,Beta,Gamma,B1,B2,B3,B4,x);
sigmaBH = BhabhaScatteringCrossSectionFnc(1,Beta,Gamma,B1,B2,B3,B4,x);
sigmaBB = BhabhaScatteringCrossSectionFnc(5,Beta,Gamma,B1,B2,B3,B4,x);

mfp = 1/(sigmaBC*n_at_C + sigmaBH*n_at_H + sigmaBB*n_at_B );
r1=(rand()/(RAND_MAX + 1.0));
dr = -mfp*log(r1);

// Bremsstrahlung
DeltaEB = (4*N*T_4*Z*(Z+1)*2.0736/(m_e*m_e*137))*(log(2*T_4/m_e)-1/3)*dr;
// e = 1.2 sqrt(MeV.fm) e^4 = 2.0736 (MeV.fm)^2

// Bethe Bloch
DeltaEBB = (pow(e2pi4epsilon,2)*(4*pi*N/(m_e*v2))*Z*(log(2*m_e*v2/0.01*Z)-
--> log(1-v2)-v2))*dr;

Q = DeltaEBB + DeltaEB;
// Q = DeltaEBB;

for (int k = 1;k < 4;k++)
{
pR_p[k-1] = pR_p[k-1] + (pV_p[k]/sqrt(v2))*dr;
RelativeR[k-1] = pR_p[k-1] - Origin[k-1];
}

Rnow = sqrt(pow(RelativeR[0],2)+pow(RelativeR[1],2)+pow(RelativeR[2],2));

for (int k = 0;k < 3;k++)
{
EnergyPosition[k] += RelativeR[k]*Q;
}
EnergyPosition[3] += Q;

if (FirstRun == 0)
{
Semi_Perimeter =(dr+R_prev+Rnow)/2; //Finding the area between the
--> Positron and Origin
// cout<<Positron_Area<<"\tSemi\t"<<Semi_Perimeter<<"\n";
// cout<<"SP = "<<Semi_Perimeter<<"\t"<<(Semi_Perimeter-dr)<<"\t"<<

```

```

--> (Semi_Perimeter-R_prev)<<"\t"<<(Semi_Perimeter-Rnow)<<"\n";
// cout<<Positron_Area<<"+"<<(Semi_Perimeter*(Semi_Perimeter-dr)*
--> (Semi_Perimeter-R_prev)*(Semi_Perimeter-Rnow)<<"\n";
Positron_Area += sqrt(Semi_Perimeter*(Semi_Perimeter-dr)*
--> (Semi_Perimeter-R_prev)*(Semi_Perimeter-Rnow));
// cout<<Positron_Area<<"\n";
}

// Angle[Counter] = acos( Dot3(pDumVec,pZaxis)/(Vec3Mag(pDumVec)));
// Distance[Counter] = Rnow;
// Qpoint[Counter] = Q;
// cout<<"Angle[] = "<<Angle[Counter]<<"\n";

FirstRun = 0;

Escape = 0;
// Escape = WallCheck(pR_p,Escape);
if (Escape == 1)
{
pPositronData[0] = 1;
break;
}

if (Rnow > Rmax)
{
Rmax = Rnow; //Finds the maximum distance from the origin
for (int i=1;i<4;i++)
{
pDumVec[i-1] = p4mom_p[i];
}
RmaxT = T_4;
ThetaMax = acos( Dot3(pPosOri4mom,pDumVec)/(Vec3Mag(pPosOri4mom)*
--> Vec3Mag(pDumVec)));
}
t = t + dr/sqrt(v2);
// PP<<t/3.0e23<<"\t"<<pR_p[0]*1e-12<<"\t"<<pR_p[1]*1e-12<<"\t"<<pR_p[2]*
--> 1e-12<<"\t"<<Q<<"\t"<<T_4<<"\n"; //Distances in mm,time in seconds

theta = TwoPartScatterFnc(p4mom_p,Q); // theta in radians

// Now rotate by theta away from the direction of the original momentum,
// then rotate around the original momentum by a random angle
for (int k = 0;k<3;k++)
{
n1[k] = pV_p[k+1]/sqrt(v2);
}

normalizer = sqrt(pow(n1[0],2)+pow(n1[1],2));
n_perp[0] = -n1[1]/normalizer;
n_perp[1] = n1[0]/normalizer;

```

```

n_perp[2] = 0;

phi =(rand()/(RAND_MAX + 1.0))*2*pi;

//      | 1+(1-cos(theta))*(pow(x,2)-1) -z*sin(theta)+(1-cos(theta))*x*y
--> y*sin(theta)+(1-cos(theta))*x*z|
//      R = | z*sin(theta)+(1-cos(theta))*x*y 1+(1-cos(theta))*(pow(y,2)-1)
--> -x*sin(theta)+(1-cos(theta))*y*z|
//      |-y*sin(theta)+(1-cos(theta))*x*z x*sin(theta)+(1-cos(theta))*y*z
--> 1+(1-cos(theta))*(pow(z,2)-1)|

n2[0] = n1[0]*(1+(1-cos(theta))*(pow(n_perp[0],2)-1)) + n1[1]*
--> (-n_perp[2]*sin(theta)+(1-cos(theta))*n_perp[0]*n_perp[1]) + n1[2]*(n_perp[2]*
--> sin(theta)+(1-cos(theta))*n_perp[0]*n_perp[2]);
n2[1] = n1[0]*(n_perp[2]*sin(theta)+(1-cos(theta))*n_perp[0]*n_perp[1])
+ n1[1]*
--> (1+(1-cos(theta))*(pow(n_perp[1],2)-1)) + n1[2]*(-n_perp[0]*
--> sin(theta)+(1-cos(theta))*n_perp[1]*n_perp[2]);
n2[2] = n1[0]*(-n_perp[1]*sin(theta)+(1-cos(theta))*n_perp[0]*n_perp[2])
+ n1[1]*
--> (n_perp[0]*sin(theta)+(1-cos(theta))*n_perp[1]*n_perp[2]) + n1[2]*(1+
--> (1-cos(theta))* (pow(n_perp[2],2)-1));

nn = sqrt(pow(n2[0],2)+pow(n2[1],2)+pow(n2[2],2));
n2[0]=n2[0]/nn; //Make sure that n2 is normalized
n2[1]=n2[1]/nn;
n2[2]=n2[2]/nn;

n3[0] = n2[0]*(1+(1-cos(phi))*(pow(n1[0],2)-1)) + n2[1]*(-n1[2]*sin(phi)+
--> (1-cos(phi))*n1[0]*n1[1]) + n2[2]*(n1[2]*sin(phi)+(1-cos(phi))*n1[0]*n1[2]);
n3[1] = n2[0]*(n1[2]*sin(phi)+(1-cos(phi))*n1[0]*n1[1]) + n2[1]*(1+(1-cos(phi))*
--> (pow(n1[1],2)-1)) + n2[2]*(-n1[0]*sin(phi)+(1-cos(phi))*n1[1]*n1[2]);
n3[2] = n2[0]*(-n1[1]*sin(phi)+(1-cos(phi))*n1[0]*n1[2]) + n2[1]*(n1[0]*sin(phi)+
--> (1-cos(phi))*n1[1]*n1[2]) + n2[2]*(1+(1-cos(phi))*(pow(n1[2],2)-1));

nn = sqrt(pow(n3[0],2)+pow(n3[1],2)+pow(n3[2],2));
n3[0]=n3[0]/nn; //Make sure that n3 is normalized
n3[1]=n3[1]/nn;
n3[2]=n3[2]/nn;

T_4 = p4mom_p[0]-m_e;
P1Mag = sqrt(p4mom_p[1]);
pV_p[0] = sqrt(1 - (1/pow(1+T_4/m_e,2)));

for (int k=1;k<4;k++)
{
p4mom_p[k] = P1Mag*n3[k-1];
pV_p[k] = pV_p[0]*n3[k-1];
}

```



```

// ET<<t<<"\t"<<T_4<<"\t"<<Q<<"\n";

// Output Original angle from z-axis, angle from original to furthest point,
--> distance between origin and furthest point,T at that point, original
--> Energy. Distances in mm, angles in degrees, energy in MeV
// ofstream PD("Data\PosDat.dat",ios::app);
// PD <<ThetaOri*180/pi<<"\t"<<ThetaMax*180/pi<<"\t"<<Rmax*1e-12<<"\t"<<RmaxT<<
--> "\t"<<E<<"\n";
Counter += 1;
// cout<<"Counter = "<<Counter<<"\n";
}

double AverageAngle = 0, AverageDistance = 0, TotalQ = 0;

// for(int j = 0;j < Counter; j++)
// {
// AverageAngle += Angle[j] * Qpoint[j];
// AverageDistance += Distance[j] * Qpoint[j];
// TotalQ += Qpoint[j];
// }
// AverageAngle = AverageAngle/TotalQ;
// AverageDistance = AverageDistance/TotalQ;

for (int k = 0;k < 3;k++)
{
// cout<<EnergyPosition[k]<<"\t";
EnergyPosition[k] = EnergyPosition[k]/EnergyPosition[3];
pDumVec[k] = EnergyPosition[k];
pR_p[k] = Origin[k] + EnergyPosition[k];
// cout<<EnergyPosition[k]<<"\t";
}
// }cout<<EnergyPosition[3]<<"\n";

AverageAngle = acos( Dot3(pDumVec,pZaxis)/(Vec3Mag(pDumVec)));
AverageDistance = Vec3Mag(pDumVec);

// cout<<"Angle = "<<AverageAngle*180/pi<<"\tDistance = "<<
--> AverageDistance*1e-12<<"\n";

pPositronData[1] = Rmax*1e-12; //in mm
pPositronData[2] = RmaxT; //in MeV
pPositronData[3] = ThetaMax*180/pi; //in degrees
pPositronData[4] = Positron_Area*1e-24; //in mm^2
pPositronData[5] = AverageAngle*180/pi; //in degrees
pPositronData[6] = AverageDistance*1e-12; //in mm
pPositronData[7] = t;

// cout<<"Rmax = "<<Rmax*1e-12<<"\t mm\tPositron Area = "<<
--> Positron_Area*1e-24<<"\tmm^2\n";
}

```


B.6 NeutronFnc

```

void NeutronMeanFreePaths(double *p_mfp,double *pV)
{
double rho = 4.75e-17; // density in fm^-3
double rho_b = 0.05*rho; //%Boron loaded

double m_red = (m_n*m_p)/(m_n+m_p);
double E = 0.5*pow(pV[0],2)*m_red;
double Sigma_b = 384000/sqrt(E/(0.025e-6));
// cout<<"Sigma_b = "<<Sigma_b<<"\n";
double hbarc = 197.3; // MeV.fm
double k = sqrt(2*m_red*E/pow(hbarc,2));
// cout<<"k = "<<k<<"\n";

double a_s = -23.715; // Scattering length in fm
double r_s = 2.73; // Effective range in fm
double a_t = 5.423; // Scattering length in fm
double r_t = 1.748; // Effective range in fm

double Sigma_s = 4*pi/(pow(k,2)+pow((1/a_s+0.5*r_s*pow(k,2)),2));
double Sigma_t = 4*pi/(pow(k,2)+pow((1/a_t+0.5*r_t*pow(k,2)),2));
double Sigma_np = (3*Sigma_t)/4 + (1*Sigma_s)/4;
// cout<<"Sigma_np = "<<Sigma_np<<"\n";
// cout<<"Sigma_s = "<<Sigma_s<<"\t";
// cout<<"Sigma_t = "<<Sigma_t<<"\n";
//
p_mfp[2] = 1/(rho*Sigma_np);
p_mfp[1] = 1/(rho_b*Sigma_b);
p_mfp[0] = 1/(1/p_mfp[2]+1/p_mfp[1]);
// cout<<"Mean free paths = "<<p_mfp[0]<<"\t"<<p_mfp[1]<<"\t"<<p_mfp[2]<<"\n";
}

void NeutronNewVelocity(double *pV)
{
double r1 = (rand()/(RAND_MAX + 1.0));
double r2 = (rand()/(RAND_MAX + 1.0));

double V_m[3],V_cm[3],magVcm;

for(int m = 0;m<3;m++)
{
V_m[m]=pV[m+1]/((m_p/m_n) + 1);
V_cm[m]=pV[m+1]-V_m[m];
}
magVcm = sqrt(pow(V_cm[0],2)+pow(V_cm[1],2)+pow(V_cm[2],2));

double phi = 2*pi*(r1);
double z = 2*r2-1;

```

B.6. NEUTRONFNC

81

```

double theta = acos(z);
double x = magVcm*sin(theta)*cos(phi);
double y = magVcm*sin(theta)*sin(phi);
z = z*magVcm;
pV[1] = x + V_m[0];
pV[2] = y + V_m[1];
pV[3] = z + V_m[2];
pV[0] = sqrt( pow(pV[1],2) + pow(pV[2],2) + pow(pV[3],2) );
}

double NeutronCapture(double *pV,double *pR,int *pNeutronData)
{
double *p_mfp,time,Tlab;
int Scatterings,Capture,Escape,Disappear;
time = 0;
pNeutronData[0] = 0;
Capture = 0;
Escape = 0;
Disappear = 0;
p_mfp = new double[3]; //[ Mean free path,Mean free path in Boron,
--> Man Free path in protons]

double Tmin = 0.025e-6; //Thermal cutoff
Tlab = 0.5*m_n*pow(pV[0],2);

while (Tlab > Tmin)
{
double r1 = (rand()/(RAND_MAX + 1.0));
double r2 = (rand()/(RAND_MAX + 1.0));

NeutronMeanFreePaths(p_mfp,pV);
double CaptureP = p_mfp[0]/p_mfp[1];
double dr = -p_mfp[0]*log(r1);

for (int i = 0;i<3;i++)
{
pR[i] = pR[i]+dr*(pV[i+1]/pV[0]); //Update postion
}

Escape = WallCheck(pR,Escape);
if (Escape == 1)
{
// cout<<"Escape!!!\n";
break;
}

time = time + dr/pV[0]; //Update time

if (r2 <= CaptureP)
{

```

```
Capture = 1;
break;
}
NeutronNewVelocity(pV); //Generates a random new velocity to
--> simulaste scatteirng off proton
pNeutronData[0] += 1;
Tlab = 0.5*m_n*pow(pV[0],2); //Update Kinetic energy
}
if (Capture == 0 && Escape == 0)
{
pNeutronData[3] +=1;
}
pNeutronData[1] +=Capture;
pNeutronData[2] +=Escape;
return time;
}
```

Bibliography

- [1] R. J. de Meijer, W. van Westrenen, *The feasibility and implications of nuclear georeactors in Earth's core-mantle boundary region*, South African Journal of Science, **104**, 111-118 (2008).
- [2] R. J. de Meijer, W. van Westrenen, *An Alternative Hypothesis For The Formation Of The Moon*, 40th Lunar and Planetary Science Conference (2009) (unpublished).
- [3] R.J. de Meijer, E.R. van der Graaf and K.P. Jungmann, Nuclear Physics News **14**, 20 - 25, (2004).
- [4] http://www.icdp-online.org/contenido/icdp/front_content.php?idcat=695. Last visited 08 October 2009.
- [5] F.D. Smit et. al., *Neutron detection, the key to direction sensitive geoneutrino detectors*, Proceedings of Science, FNDA, **96** (2006).
- [6] R. J. de Meijer et. al., *Earth, Moon and Planets*, **99**, 193-206, (2006).
- [7] H. Nunokawa, W. J. C. Teves, R. and Zukanovich Funchal, JHEP, **0311**, 020 (2003) ,arXiv:hep-ph/0308175 v2 5 Nov 2003.
- [8] G. Fiorentini, F. Mantovani and B. Ricci, Phys. Let. B **557**, 139-146, (2003).
- [9] Fabio Mantovani, Luigi Carmignani, Gianni Fiorentini, Marcello Lissia, Phys. Rev. D **69**, 013001 (2004), arXiv:hep-ph/0309013 v2 27 Nov 2003.
- [10] K.A. Hochmuth et. al., *Earth Moon and Planets* **99**, 253-264 (2006) ,arXiv:hep-ph/0610048v1 4 Oct 2006.
- [11] R. S. Raghavan, arXiv:hep-ex/0208038v2.

- [12] <http://www-pub.iaea.org/mtcd/publications/PubDetails.asp?pubId=6120>.
Last visited 5 October 2009.
- [13] F. Suekane et. al., *Tsukuba 2003*, Scintillating crystals, 279-290,
arXiv:physics/0404071.
- [14] Byron D. Dieterle, *The CHOOZ and KamLAND Reactor Neutrino Experiments*.
- [15] E. Benedito, J.M. Fernández-Varea, F. Salvat, *Nucl. Instr. and Meth. B*,
174,91-110, (2001).
- [16] <http://www.particleadventure.org/>. Last visited 13 February 2010.
- [17] V.D. Rusov et. al., arXiv:hep-ph/0312296v2.
- [18] W. R. Nelson, H. Hirayama and D. W. O. Rogers, *SLAC - 265* (1985).
- [19] ICRU Report, **49** (1993).
- [20] G. F. Knoll, *Radiation Detection and Measurement*, 3rd edition, John Wiley & Sons, Inc., (2000).
- [21] R. Fearick, Private communication.
- [22] V.D. Rusov et. al., arXiv:nucl-th/0605025v2.
- [23] Craig S. Levin and Edward J. Hoffman, *Phys. Med. Biol.* **44**, 781-799 (1999).
- [24] F. D. Smit, Private communication.