SIMULATING RADIOACTIVE DECAY

DEVELOPMENT OF ALGORITHM:

The heart of the algorithm to solve this problem computationally is pretty straightforward and is provided in the assignment. The code "RadioactiveDecay.cpp" is executable on UNIX machine in ROOT session only and is capable of solving two cases each for part-a and part-b of given question in one run and saves all the histograms automatically in "RadioactiveDecay.root" under two sub-directories "PartA" and "PartB" respectively. We will discuss the development of code in following passages.

We declared header-files from C++ and ROOT libraries and adopted "using namespace std" convention for the code. Besides RadioactiveDecay() which is our main function of type "int", we have constructed poissonf(), expo_s() and Pol_1(), all of type "double" and serve us for fitting purpose in ROOT.

Part A:

We introduced a "do-while()" loop and utilize "switch" with "case 1" and "case 2", each solving one part of the question for us. Upon execution, the code directly enters "case 1", initializes N_o=100 and α=0.01/sec and enters the time-step-loop. At each time step of one second, it runs a secondary loop generating random numbers between [0,1] as many times as the parent nuclei are present at the start of the secondary loop. In this secondary loop, it counts the instances when random number generated is smaller than probability of a nucleus to undergo a radioactive decay, thus counts the total nuclei decayed in one second. After secondary loop is executed, the total decayed nuclei in that time-step are subtracted from number of nuclei present at the beginning that particular time-step, the remaining nuclei are stored in 1D histogram and next time-step is initiated with remaining nuclei as input with total nuclei decayed is again set to zero. This procedure is repeated until last time-step is executed and we write histogram for N_o=100 and α=0.01/sec, fit our histogram using expo_s() and Pol_1() with red color and plot on it the theoretical curve in green color. The resulting histograms are shown in figure 1.1 with description.

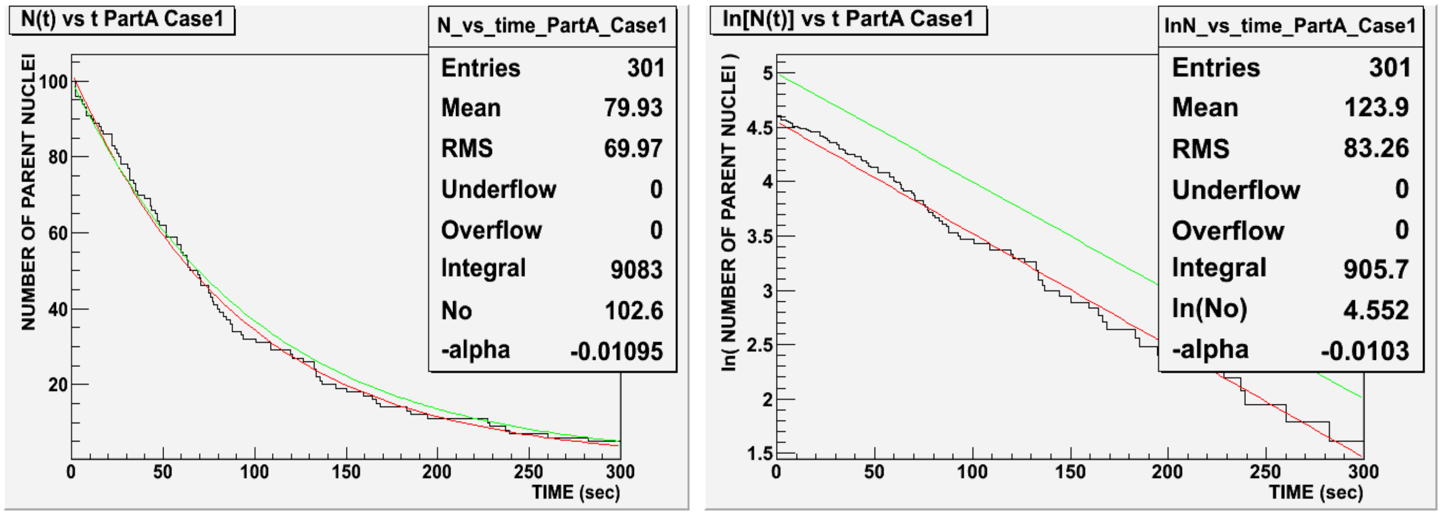


Figure 1.1: N(t) vs Time on Left and ln(N(t)) vs Time on Right for N_o=100 and α=0.01/sec. Red curve is best fit to data and green curve is theoretical expectation.

We change the initialization to N_o=5000 and α=0.03/sec for case 2 after we save histograms for N_o=100 and α=0.01/sec and repeat the whole procedure to obtain histograms for case 2. The obtained histograms are shown in figure 1.2 with description. The exponential best fit and theoretical curve are in good agreement in this particular case.

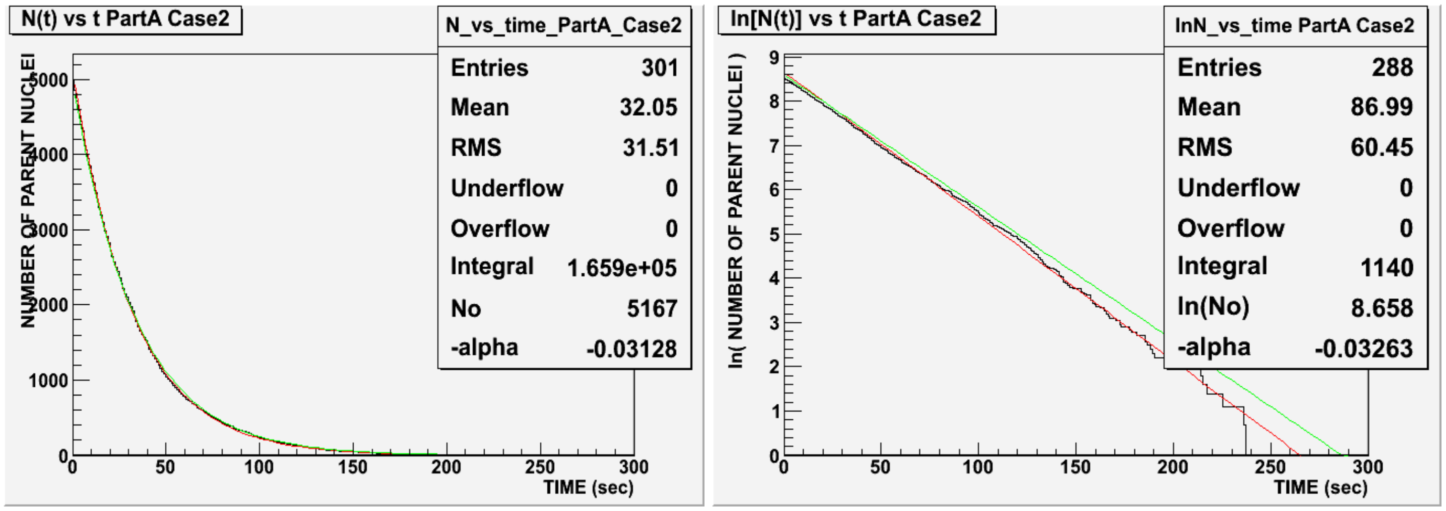


Figure 1.2: N(t) vs Time on Left and ln(N(t)) vs Time on Right for N_o=5000 and α=0.03/sec. Red curve is best fit to data and green curve is theoretical expectation.

Part B:

Once we reach towards end of "case 1", a parameter "QUESTIONPART" is given an integer value 2 and hence, the do-while() loop switches to "case 2" which solves part-b for us. The algorithm is typically identical to part a, with only difference being the total nuclei decayed is not set to zero at the beginning of each new time-step, in this case 10 seconds each. In fact, we initiate total nuclei decayed at beginning of each sample simulation and plot the aggregate of it to histogram after we hit total time of 100 seconds. We simulate 1000 experiments and store total decayed nuclei for each experiment in histogram. At end of 1000 experiments, we write the histograms and fit Poisson Distribution utilizing user-defined function, thanks to ROOT.

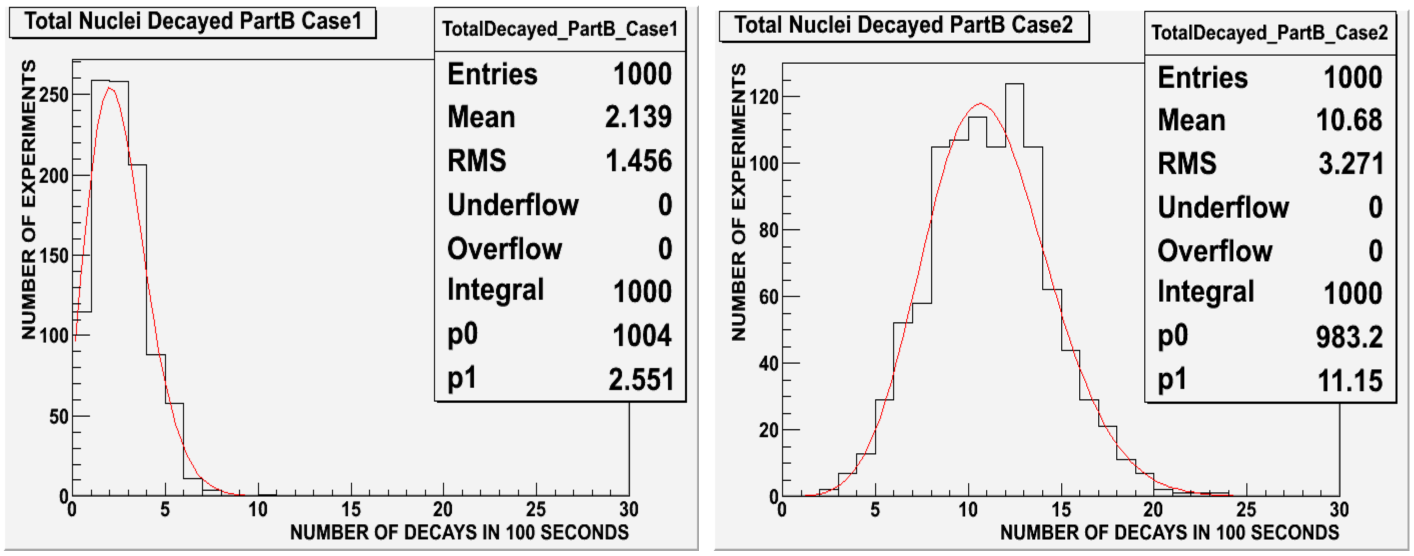


Figure 1.3: Accumulative result of 1000 experiments observing Total Nuclei Decayed in 100 seconds with time-step of 10 seconds and Poisson Fit. On left side, N_o=500 and α=4.0E-5/sec is simulated and on right side, N_o=500 and α=2.0E-4/sec is histogrammed.

Limit On Δt:

As we know that ΔN=-NαΔt → (ΔN/N)=-αΔt, we should choose our time-step Δt such that product of Δt and α should approximate to ratio of nuclei decayed to nuclei present just before time-step.

The time-step should not be comparable with mean-life of the radioactive sample. Instead, it should be very small in comparison of mean-life.

Last but not the least, we realize that we compare *p= αΔt* to a random number λ ∈ [0,1] in our simulation, with α being the decay constant of particular radioactive nuclei, we are left with only liberty to choose Δt such that the product αΔt should fall in the range [0,1]. For practical goodness, αΔt≈0.3 is a decent choice for simulation.

IMPLEMENTING REJECTION AND TRANSFORMATION TECHNIQUES

DEVELOPMENT OF ALGORITHM:

The function provided in the problem is;

$$f(x) = \frac{1}{\sin^2\theta + a\cos^2\theta} \text{ with Case 1: } a = 0.001 \text{ and Case 2: } a = 0.5$$

and we are asked to plot this function utilizing Acceptance-Rejection Technique and Transformation Technique. We observed that the function is even about zero and has a period of [-π/2, π/2] or [0, π]. The key requirement in order to carry any of these techniques is that we should have a probability density function (pdf) before hand. A simple numeric integration of the given function utilizing trapezoidal rule reveals that the function is not normalized and hence not a pdf for a =0.001 and a =0.5;

$$\int_{-\pi/2}^{\pi/2} f(\theta)d\theta = \int_{-\pi/2}^{\pi/2} \frac{1}{\sin^2\theta + a\cos^2\theta} d\theta = 99.34588 \text{ for } \alpha = 0.001$$

$$\int_{-\pi/2}^{\pi/2} f(\theta)d\theta = \int_{-\pi/2}^{\pi/2} \frac{1}{\sin^2\theta + a\cos^2\theta} d\theta = 4.44288 \text{ for } \alpha = 0.5$$

which promptly gives us the normalization constant for both cases. Here, we take the liberty to choose [-π/2, π/2] as our working domain.

Normalization Constant, NC = 1/99.34588 for a =0.001

Normalization Constant, NC = 1/4.44288 for a =0.5

Once, we have obtained a pdf, we are good to move a step further and try implementing any of two techniques.

Acceptance-Rejection Technique:

The algorithm for Acceptance-Rejection Technique involves two steps. For first step, we generate a θ_{trial} ∈ [-π/2, π/2] through following equation which involves a random number as well;

$$\theta_{trial} = -\frac{\pi}{2} + \left(\frac{\pi}{2} - \left(-\frac{\pi}{2}\right)\right) \times RandomNumber1$$

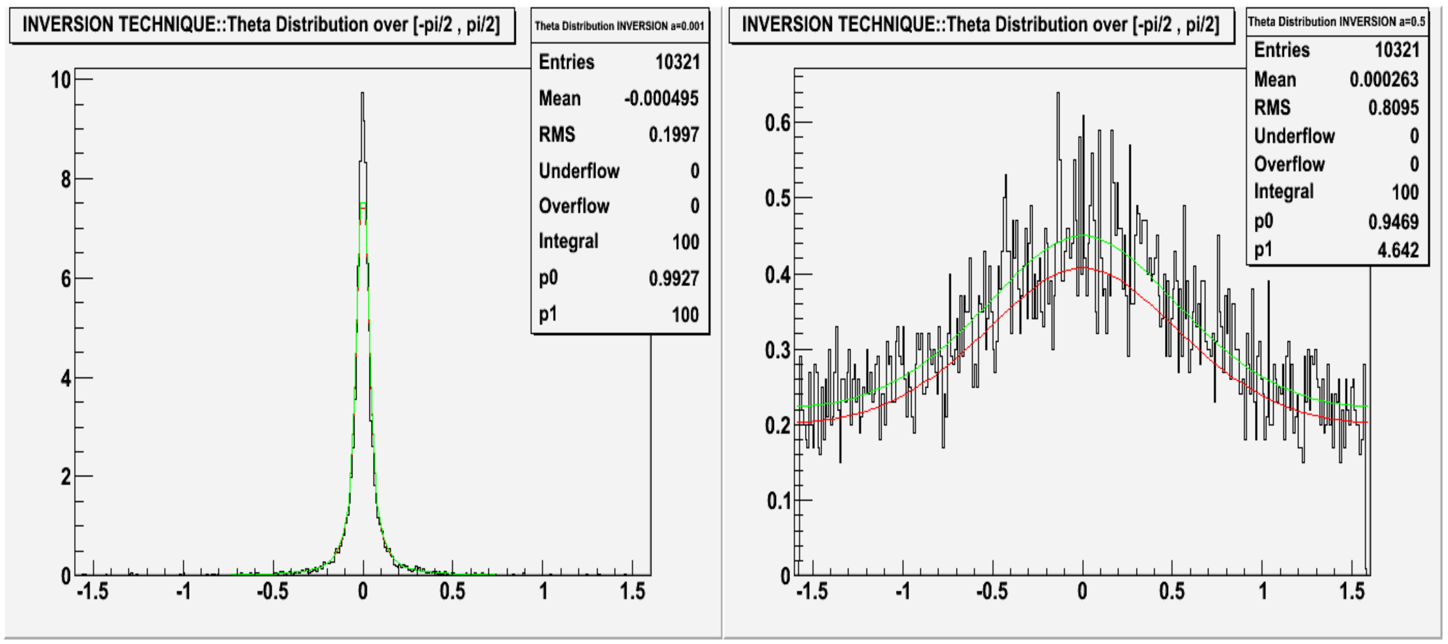
And second step involves selection or rejection of θ_{trial} based upon evaluation of pdf at θ_{trial} and comparing it with Maximum value of pdf over [-π/2, π/2] times freshly generated second random number;

$$\text{Accept } \theta_{trial} \text{ if } \frac{NC}{\sin^2\theta_{trial} + a\cos^2\theta_{trial}} > RandomNumber2 \times \frac{NC}{\alpha}$$

Where $\frac{NC}{\alpha}$ is the maximum value of pdf at θ=0° over [-π/2, π/2].

We make use of "do-while()" loop to collect 10K events as we need to generate well over 10K events because of rejection criteria involved in the algorithm. Accepted θ_{trial} , 10K in number are written to histogram which is not normalized to unit area at the moment. We call a user defined function SolveForTheta_REJECTION() in our main algorithm to calculate one side of inequality for step two. Using functions make a code versatile for general use in future besides its convenient to understand as well.

We normalize 1D histograms to unit area by dividing contents of each bin with product of individual bin width and total number of events(10K in our case) and plot over it the expected curve in green color and the best fit curve in red color. The histograms obtained are presented in figure 2.1 for both values of α.



<u>Figure 2.1</u>: Normalized Theta distribution in Radians using Acceptance-Rejection Technique. Left Histogram is for Case α=0.001 and Right Histogram is for Case α=0.5. Red Curve is the best fit and Green Color is the expected curve.

Transformation Technique:

The implementation of transformation technique requires only one step and that is analytical evaluation of a definite integral of pdf from lower limit to some value θ Є [Lower Value, Upper Value], equating it to some random number generated and retrieving θ from that equation. Repeating this process to reasonable number of times gives us the distribution.

We start again on our particular problem;

$$\int_{-\pi/2}^{\theta} PDF d\theta = \int_{-\pi/2}^{\theta} \frac{NC}{\sin^2\theta + a\cos^2\theta} d\theta \Rightarrow NC \int_{-\pi/2}^{\theta} \frac{\sec^2\theta}{\frac{\sin^2\theta}{\cos^2\theta} + a} d\theta \Rightarrow NC \int_{-\pi/2}^{\theta} \frac{\sec^2\theta}{\tan^2\theta + a} d\theta$$

We make use of substitution $x = \tan\theta \Rightarrow dx = \sec^2\theta d\theta$ and our integral is solvable using standard integration formula;

$$NC \int_{\tan\left(-\frac{\pi}{2}\right)}^{\tan\theta} \frac{1}{x^2 + \left(\sqrt{a}\right)^2} dx \Rightarrow \left. \frac{NC \tan^{-1}\left(x/\sqrt{a} \right)}{\sqrt{a}} \right|_{\tan\left(-\frac{\pi}{2}\right)}^{\tan\theta} \Rightarrow \left. \frac{NC \tan^{-1}\left[\tan\theta/\sqrt{a} \right]}{\sqrt{a}} \right|_{-\pi/2}^{\theta}$$

Equating it to a random number, we finally have;

$$\theta = \tan^{-1}\left[\sqrt{a} \tan\left\{ \frac{\sqrt{a}}{NC}\left[Y + \frac{NC}{\sqrt{a}} \tan^{-1}\left\{ \frac{\tan\left(-\pi/2\right)}{\sqrt{a}} \right\} \right] \right\} \right]$$

Where Y is a random number between 0 and 1 inclusive.

NUMERICAL LIMITATION:

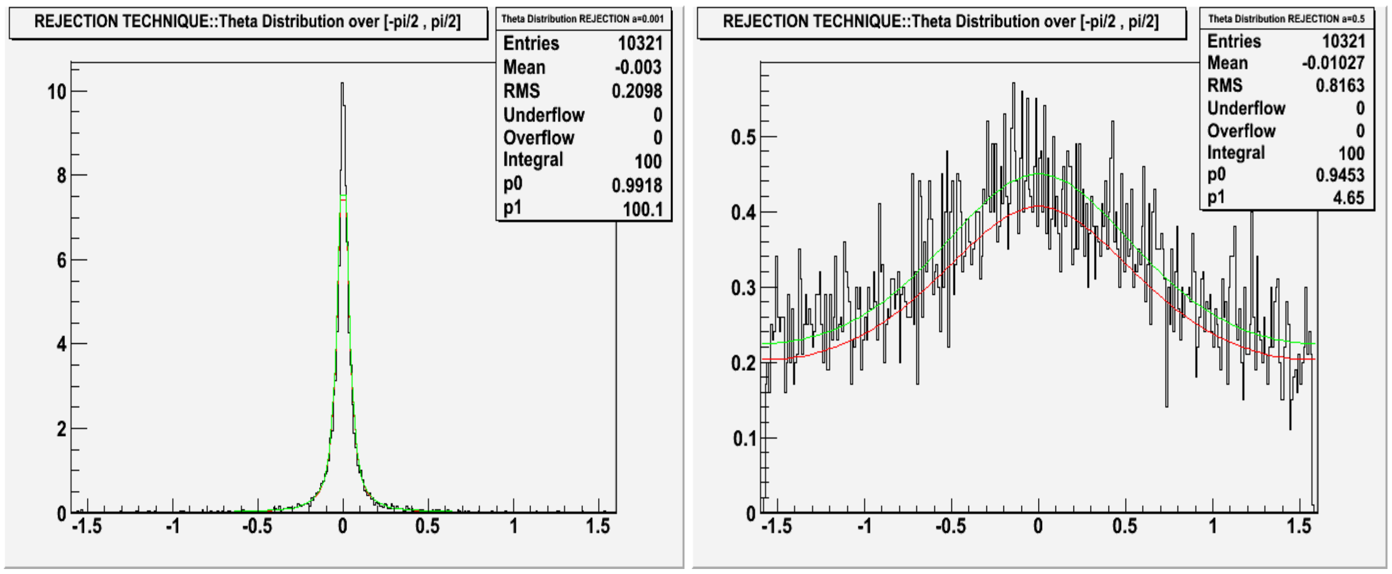
We find that selection of interval [-π/2, π/2] involves numeric evaluation of tan(-π/2) in function SolveForTheta_INVERSION() which happens to be $-\infty$. Apparently it seems that θ cannot be calculated numerically in code and that Inversion technique is not applicable over selected interval, however we find a way around by realizing that;

$$\tan^{-1}\left\{ \frac{\tan\left(-\pi/2\right)}{\sqrt{a}} \right\} = -\frac{\pi}{2}$$

And hence, SolveForTheta_INVERSION() returns θ, evaluated using following equation;

$$\theta = \tan^{-1}\left[\sqrt{a} \tan\left\{ \frac{\sqrt{a}}{NC}\left[Y + \frac{NC}{\sqrt{a}}\left(-\frac{\pi}{2} \right) \right] \right\} \right]$$

We collect 10K events in 1D histogram and normalize our histograms once again by dividing each bin content with a product of individual bin content and total number of events generated. We also apply best fit to the data in red color and expected curve is in green color.



<u>Figure 2.1</u>: Normalized Theta distribution in Radians using Acceptance-Rejection Technique. Left Histogram is for Case α=0.001 and Right Histogram is for Case α=0.5. Red Curve is the best fit and Green Color is the expected curve.

<u>CPU Time Consumed:</u>

We include "Tsystem.h" and "TStopWatch.h" ROOT libraries to our code and utilize Start() and Stop() methods to calculate CPU time consumed for two techniques for α=0.001 and α=0.5. We tabulate the time consumed at one of THUNER-GW5's node in Table 2.1.

TECHNIQUE	**α=0.001**	**α=0.5**
ACCEPTANCE-REJECTION	1.75 sec	0.09 sec
TRANSFORMATION	0.04 sec	0.04 sec

<u>TABLE 2.1:</u> CPU Time Utilized At THUNER-GW5 Node.

This concludes our analysis for Acceptance-Rejection and Transformation Techniques. We can also produce results using same code for given function over interval [0, 2π]. The only difference is Normalization Constant which will be different for given interval.

SIMULATING COMPTON SCATTERING AND CALCULATING CROSS-SECTION USING KLEIN-NISHINA FORMULA

ALGORITHM DEVELOPMENT:

We make use of Importance Sampling MC Technique to simulate Compton Scattering Phenomenon. The equations given to us are in natural units, therefore we develop the code accordingly. The code provides the user with four different options to select scale of energy and asks for incident photon energy and total number of Compton Events to be generated.

The main function of our code is ComptonScattering() of type "int". After we are done with declaring/initializing variables and histograms, we develop a do-while() loop in the program that performs core calculations for us. Inside this do-while() loop, a random number λ is generated and is passed on to SolveForU() along with initial photon energy k. SolveForU() is a user defined function of type "double" that solves the following equation;

$$u = \frac{m}{k}\left[\left(1+2\frac{k}{m}\right)^{\lambda} - 1\right] \text{ where } u = (1-\cos\theta)$$

This equation is provided to us and generates u based on Inversion Technique of Approximate Function. Once, u is generated and its value is returned by SolveForU(), we calculate θ in radian.

With known u, we calculate weight w which is the essence of Importance Sampling Technique. The underlying idea and governing equations, specific to our case are presented here. Klein-Nishina Formula is given by the following equation;

$$\sigma(\theta,\phi)\,d\theta d\phi = \frac{\alpha^2}{2m^2}\left[\left(\frac{k'}{k}\right)^3 + \left(\frac{k'}{k}\right) - \left(\frac{k'}{k}\right)^2 \sin^2\theta\right]\sin\theta\, d\theta d\phi$$

It is difficult to apply Transformation Technique or Acceptance-Rejection Technique to the above equation to generate θ distribution. Therefore, we define an approximate function for Klein-Nishina Formula as followed;

$$\sigma^{approx}(\theta,\phi)\,d\theta d\phi = \frac{\alpha^2}{2m^2}\left(1+\frac{k}{m}u\right)^{-1}\sin\theta\, d\theta d\phi$$

And the weight w is calculated using following relationship;

$$\sigma(\theta,\phi) = w\sigma^{approx}(\theta,\phi) \;\Rightarrow\; w = \frac{\sigma(\theta,\phi)}{\sigma^{approx}(\theta,\phi)} \;\Rightarrow$$

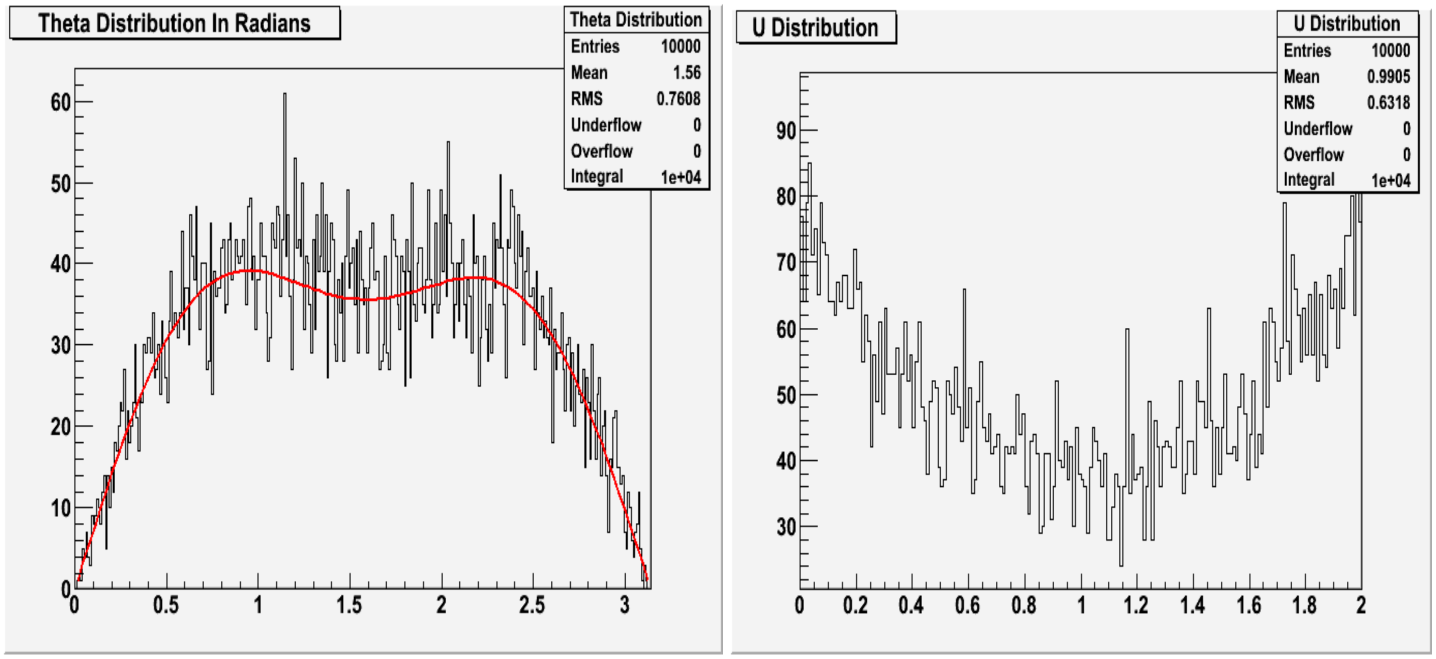
$$\Rightarrow w = \frac{\left(\frac{k'}{k}\right)^3 + \left(\frac{k'}{k}\right) - \left(\frac{k'}{k}\right)^2 \sin^2\theta}{\left(1+\frac{k}{m}u\right)^{-1}} \quad \text{as} \quad \frac{k'}{k} = \left(1+\frac{k}{m}u\right)^{-1} \quad \Rightarrow w = \left[\left(\frac{k'}{k}\right)^3 + \left(\frac{k'}{k}\right) - \left(\frac{k'}{k}\right)^2 \sin^2\theta\right]\frac{k}{k'}$$

And finally;

$$w = \left[\left(\frac{k'}{k}\right)^2 + 1 - \left(\frac{k'}{k}\right)\sin^2\theta\right] \Rightarrow w = \left[\left(\frac{k'}{k}\right)^2 + 1 + \left(\frac{k'}{k}\right)u^2 - 2u\frac{k'}{k}\right]$$

We make use of the last equation to calculate weight w in our core calculation. We have defined a function k_prime_BY_k() of type "double" to assist us calculating w efficiently.

Once the weight w is calculated, we generate a new random number λ_2 and if $w > \lambda_2 w_{\max}$, we accept the weight w, corresponding u and θ as part of our distribution function. We write three quantities in respective histograms and repeat the iteration until we collect desired number of events to be generated using do-while loop. The distribution of θ obtained in radian is the required plot. Our next step is to compare Klein-Nishina Formula and for this purpose, we have defined a function Klein_Nishina() of type "double", scale it using Fit method in ROOT and plot it alongside θ distribution obtained using Importance Sampling Technique. The θ distribution for four cases are presented in following figures.



<u>Figure 3.1:</u> Simulation of 10K Compton Scattered Events for 5KeV Incident Photon. On left side, θ distribution is presented along with Total Klein-Nishina Formula in Red Curve. On right side, u distribution is presented.

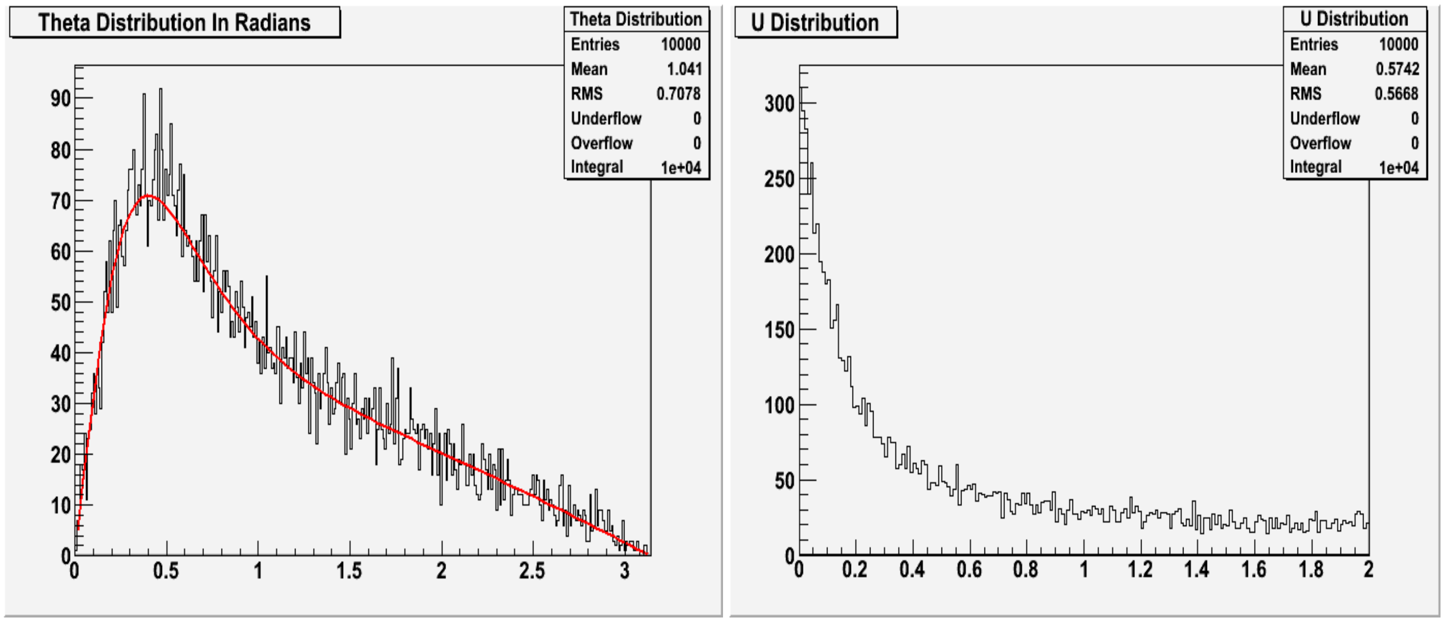


Figure 3.2: **Simulation of 10K Compton Scattered Events for 2MeV Incident Photon. On left side, θ distribution is presented along with Total Klein-Nishina Formula in Red Curve. On right side, u distribution is presented.**

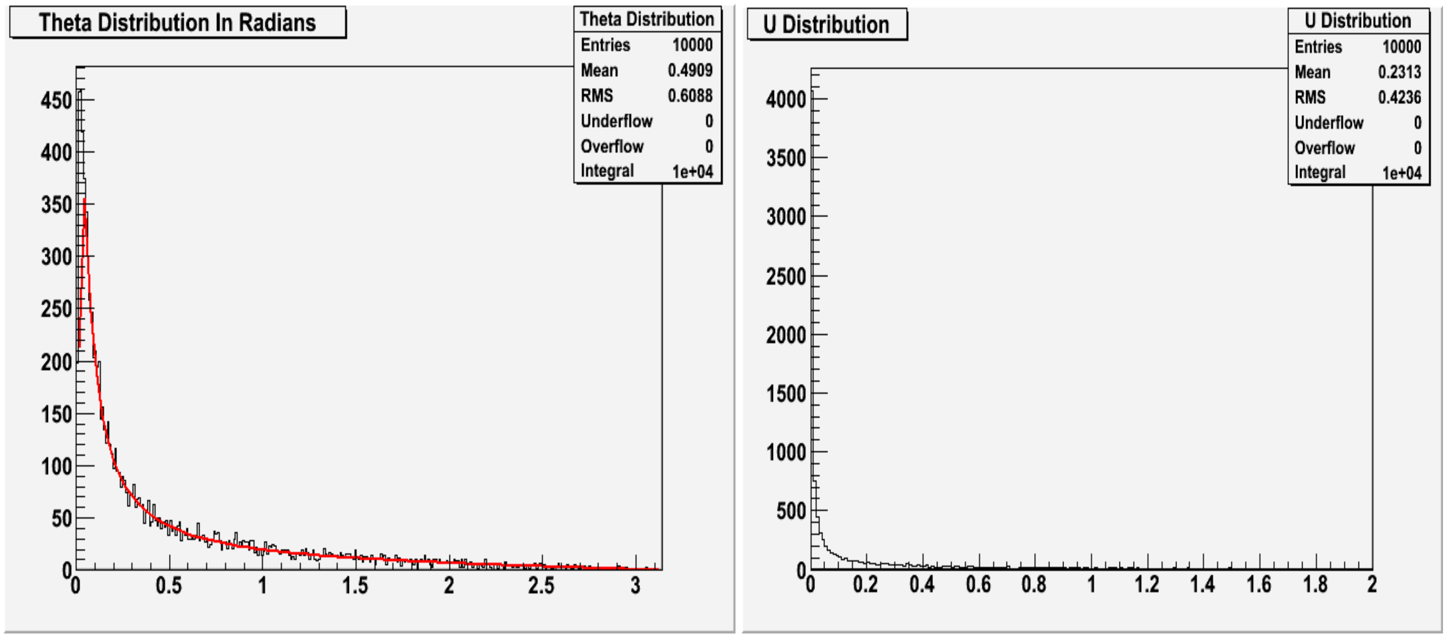
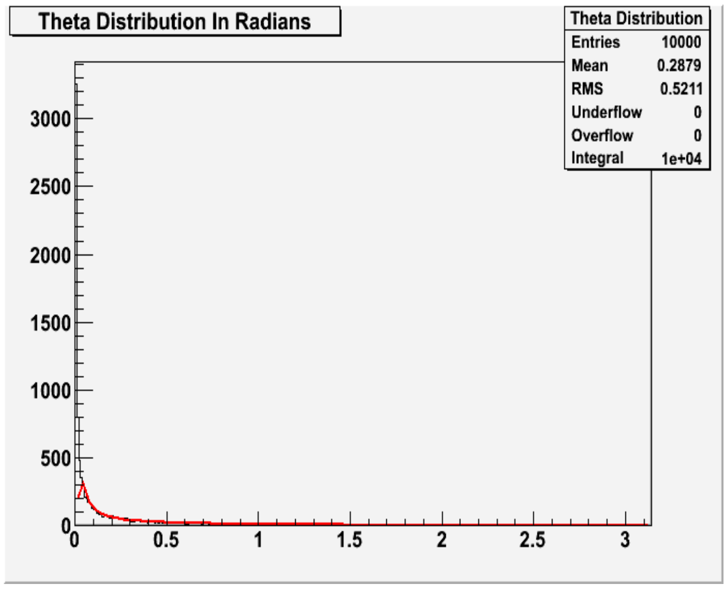


Figure 3.3: **Simulation of 10K Compton Scattered Events for 1GeV Incident Photon. On left side, θ distribution is presented along with Total Klein-Nishina Formula in Red Curve. On right side, u distribution is presented.**



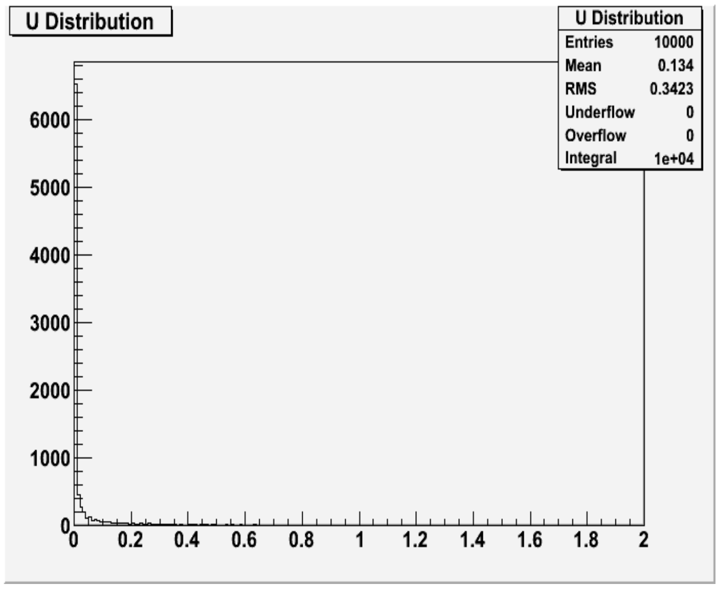


Figure 3.4: Simulation of 10K Compton Scattered Events for 1TeV Incident Photon. On left side, θ distribution is presented along with Total Klein-Nishina Formula in Red Curve. On right side, u distribution is presented.

Calculation of Total Cross-Section:

Mathematics is set up to calculate total cross-section. We know that;

$$\sigma(\theta,\phi) = w\sigma^{approx}(\theta,\phi) \;\Rightarrow\; \int \sigma(\theta,\phi)\,d\theta d\phi = \int w\sigma^{approx}(\theta,\phi)\,d\theta d\phi \;\Rightarrow\; I = I^{approx} < w >$$

Where, $< w >$ is the average weight and we use GetMean() method provided in ROOT to have its numeric value, I and I^{approx} are Total Cross-Section and Approximate Total Cross-Section respectively. It is possible that we calculate I^{approx} analytically but before we start doing so, we change expression of $\sigma^{approx}(\theta,\phi)$ from natural units to SI units.

$$\sigma^{approx}(\theta,\phi)\,d\theta d\phi = \frac{\hbar}{2m^2c^2}\left(\cdots \frac{k}{mc^2}u\right)^{-1} du d\phi$$

Integrating over solid angle, we get;

$$\iint \sigma^{approx}(\theta,\phi)\,d\theta d\phi = \iint \frac{\hbar}{2m^2c^2}\left(\cdots \frac{k}{mc^2}u\right)^{-1} du d\phi$$

$$I^{approx} = 2\pi\int_0^2 \frac{\hbar}{2m^2c^2}\left(\cdots \frac{k}{mc^2}u\right)^{-1} du$$

$$I^{approx} = 2\pi \frac{\hbar}{2m^2c^2} \int_0 \frac{1}{1+\frac{k}{mc^2}u} du$$

Making a substitution $x = 1 + \frac{k}{mc^2}u \Rightarrow dx = \frac{k}{mc^2}du$ and using standard integration formula $\int \frac{1}{x}dx = \ln(x)$, we get;

$$I^{approx} = 2\pi \frac{\hbar}{2mk} \left[\ln\left(2m\left(mc^2 + ku\right)\right) \right]_0^2$$

$$I^{approx} = 2\pi \frac{\hbar}{2mk} \left[\ln\left(2m\left(mc^2 + 2k\right)\right) - \ln(2m^2c^2) \right]$$

We make use of above analytical expression in our code to calculate I^{approx}, multiply it with $<w>$ and obtain Total Cross-Section I.

Comparison With Wolfram Demonstration Project:

We compare the total cross-section for first three cases obtained from our numeric calculation with one of the online resource simulation provided by Wolfram's Mathematica Demonstration Project at http://demonstrations.wolfram.com/KleinNishinaFormulaForPhotonElectronScattering/. The comparison is presented in Table 3.1 along with calculated uncertainty.

Uncertainty:

We calculate the uncertainty in Total Cross-Section in our code using following formula;

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} w_i^2 - \left(\frac{1}{N}\sum_{i=1}^{N} w_i\right)^2$$

Incident Photon Energy	Total Cross-Section (barn)	Wolfram Total Cross-Section (barn)	Uncertainty
5KeV	0.6848	0.6353	9.628E-02
2MeV	0.1578	7.941E-02	1.151E-01
1GeV	1.142E-03	1.111E-03	3.958E-02
1TeV	2.031E-06	NOT AVAILABLE	2.516E-02

Table 3.1: Total Cross Section for four cases are presented in barn and uncertainty is tabulated for each case.

This concludes our analysis for simulation of Compton Scattering and distribution of scattering angle governed by Klein-Nishina Formula.