Monte Carlo Technique

Geant4 and its Application to HEP and Astrophysics

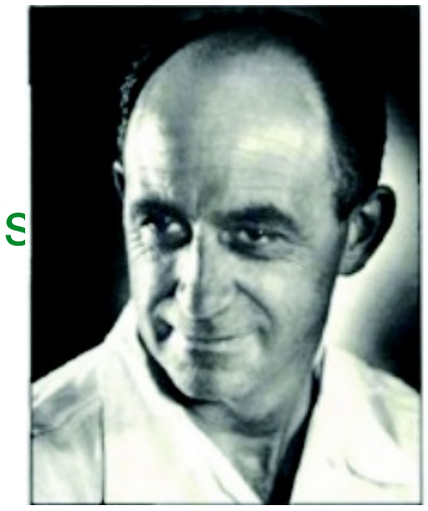
December 5, 2022

Sunanda Banerjee

The Beginning

- There was extensive research all over the world during the ’30s and ’40s that went into the creation of the atomic bomb
  - nuclear fission discovered in Germany led to the idea of using this in generating a large amount of energy
  - though a bad fallout of this is the process of destruction of many human lives, it also opened up many new ideas
    - how to predict the outcome of a process before doing the actual experiment
    - it was simultaneously invented by two groups of mathematicians and physicists
      - At the Manhattan Project, it was the work of four mathematicians: Stanislaw Ulam, John Von Neumann, Robert D. Richtmeyer, Nicholas Metropolis
      - At Rome and Chicago, it was the idea of the physicist Enrico Fermi
    - This also led to the start of the first digital computer

History



- “After spending a lot of time trying to estimate success by combinatorial calculations, I wondered whether a more practical method might be to lay it out say one hundred times and simply observe and count the number of successful plays” - S. Ullam playing Solitaire while recovering from an illness during 1946
- 1943 saw the first electro-mechanical computers solving non-linear equations. Ulam’s idea led von Neumann to think of using this new device to carry out the statistical sampling. He together with Richtmeyer worked on “Statistical Methods in Neutron Diffusion” in 1947
- The first calculation was done on the ENIAC computer in 1948, with the code finalised by December 1947
- Metropolis christened the name “Monte Carlo” and gave rise to MANIAC — Mathematical and Numerical Integrator and Computer
- Enrico Fermi had the idea even 15 years earlier than Ulam and astonished his Roman colleagues by accurately predicting experimental results using statistical techniques. He created an analogue machine FERMIAC to study neutron transport while awaiting the arrival of ENIAC

Monte Carlo Technique

- Solution of a problem as a parameter of a hypothetical population and constructing a sample of the population to obtain estimates of the parameter
  - ==> use random numbers to construct a sample
- This technique is particularly useful to study particle scattering and absorption because it involves
  - random processes (interaction, scattering, …)
  - complicated multi-dimensional integration
- The Monte Carlo technique is the ideal way to carry out multi-dimensional integration — the fastest method when the number of dimensions exceeds 3 or 4.
- We may need to carry out an integration:

$$I = \int_0^1 dx_n \cdots \int_0^1 dx_1 F(x_1, \cdots x_n)$$

  then with a set of random numbers $x_1, \cdots x_n$ in the range 0 — 1, determine F and this F will be an unbiased estimator of I
- Repeat this estimate a large number of times and the mean value of F, <F> will converge to the value of I

Random Variables

- Random variables are characterised by
  - it can have more than one value (generally any value within a range)
  - one cannot predict in advance which value it will take
  - the distribution of the variable may be well known
- Distribution of a random variable ==> probability of having a specific value
- The probability distribution function is given by

$$g(u)du \;=\; P[u < u' < u + du]$$

- Integrated distribution functions:

$$G(u) \;=\; \int_{-\infty}^{u} g(u')du'$$

$$g(u) \;=\; \frac{dG(u)}{du}$$

- G(u) increases monotonically with u. The normalisation of g is determined by

$$\int_{-\infty}^{\infty} g(u)du \;=\; 1$$

Type of Random Variables

Truly random numbers

- A sequence of truly random numbers is completely unpredictable and hence irreproducible
- Can be generated only through random physical processes (radioactive decay, arrival time of cosmic ray particles, …)
- Not so useful in the standard integration process

Quasi-random numbers:

- The sequence does not appear random (high degree of correlation) but provides the right answers to Monte Carlo integration
- They use strict mathematical formulas and provide fast convergence of certain integration
- These are of limited use

Pseudo-random numbers:

- A sequence generated according to a strict mathematical formula, but indistinguishable from a sequence generated truly randomly
- Most simulation programs use pseudo-random numbers. The heart of a simulation process is the generation of "Uniform Deviates": random numbers which lie within a specific range (0 to 1) with any number just as likely as the other

Pseudo random numbers

- start with a number of r digits. The first random number can be the middle r/2 digits. Square this number and again take the middle r/2 digits for the next random number and so on. This procedure is machine-dependent, has a large correlation and also has a small period

- Multiplicative Linear Congruential Generator (MLCG): This is the most commonly used random number generator which generates a sequence of integers between 0 and m—1 (a large number) using the recursion relation:

$$r_i = \mathrm{mod}(a \cdot r_{i-1} + b, m)$$

where a is the multiplier, b is an additive constant, r's are the random numbers in the sequence, and m is the modulus

This is very fast and transportable with a proper choice of a, b, and m. But it is not free from sequential correlation and has a short period (at most can have a period of m)

Lower-order bits are much less random than the higher-order bits. So for the choice of a random number in the range between 1 and 10, it is better to use $1 + \mathrm{int}(10. * r_i)$ rather than $1 + \mathrm{mod}(\mathrm{int}(100000000 * r_i), 10)$.

This can be improved by first making a table of random numbers generated using MLCG and then picking randomly from this table.

Pseudo random numbers

- Subtraction Method: Subtraction of two randomised numbers provide transportable random numbers of rather a large period. This is the most commonly used method in many applications:
  - Initialize a table in slightly random order with numbers that are not strictly random
  - Randomize them by subtracting a number not especially random
  - Take the difference between two numbers in the table which are apart
  - Update the table position with this number
  - Go to the next sequence of the table
- Geant4 makes use of the Random number generators defined within the package CLHEP
  - the earlier versions made use of the ranecu algorithm
  - a more recent version utilises mixmax algorithm

```
double RanecuEngine::flat() {
   const int index = seq;
   long seed1 = table[index][0];
   long seed2 = table[index][1];

   int k1 = (int)(seed1/ecuyer_b);
   int k2 = (int)(seed2/ecuyer_e);
   seed1 = ecuyer_a*(seed1-k1*ecuyer_b)-k1*ecuyer_c;
   if (seed1 < 0) seed1 += shift1;
   seed2 = ecuyer_d*(seed2-k2*ecuyer_e)-k2*ecuyer_f;
   if (seed2 < 0) seed2 += shift2;

   table[index][0] = seed1;
   table[index][1] = seed2;

   long diff = seed1-seed2;
   if (diff <= 0) diff += (shift1-1);
   return (double)(diff*prec);
}
```

- With the constants:

```
static const int ecuyer_a = 40014;
static const int ecuyer_b = 53668;
static const int ecuyer_c = 12211;
static const int ecuyer_d = 40692;
static const int ecuyer_e = 52774;
static const int ecuyer_f = 3791;
static const int shift1   = 2147483563;
static const int shift2   = 2147483399;
```

Arbitrary probability density

- To generate random numbers according to an arbitrary normalised probability density function F(x), there are several methods available. One has to choose the most efficient approach

- Transformation method:

  Let there be a random function which can produce a variable x uniformly in the range from 0 to 1: $0 \leq x \leq 1$

  This implies that the probability of generating a number between x and x+dx:

$$\begin{aligned} g(x)dx &= dx \quad \text{for } 0 \leq x \leq 1 \\ &= 0 \quad \text{for } 0 > x \text{ or } x > 1 \end{aligned}$$

  and the probability density function is normalised:

$$\int_{-\infty}^{\infty} g(x)dx = 1$$

  Now we would like to generate random numbers y with a probability density function f(y) which also follows:

Transformation Method

$$\int f(y)dy = 1$$

Using the fundamental transformation law of probability

$$dx = f(y)dy$$

$$\Rightarrow x = \int f(y)dy + c$$

So one gets

$$|f(y)dy| = |g(x)dx|$$

$$\text{or } f(y) = g(x)\left|\frac{dx}{dy}\right|$$

This is to be inverted to get

$$y = y(x)$$

Transformation Method (an example)

Let us generate an exponential distribution as an example:

$$\begin{aligned} f(y)dy &= a \cdot \exp(-a \cdot y)dy \\ \text{with } y &\geq 0 \text{ and } a > 0 \end{aligned}$$

This satisfies

$$\begin{aligned} \int_0^\infty f(y)dy &= 1 \\ \Rightarrow x &= -\exp(-a \cdot y) + c \end{aligned}$$

Imposing boundary conditions, namely $x = 0$ at $y = 0$ and x = 1 at $y = \infty$, one obtains c = 1

Thus

$$y = -\frac{1}{a}\ln(1 - x)$$

Generalised Transformation Method

The transformation method can be generalised to more than one dimension:

$x_1, x_2, \cdots$ are random deviates with joint probability distributions

$g(x_1, x_2, \cdots)\, dx_1\, dx_2 \cdots$

$y_1, y_2, \cdots$ each is a function of all x's (number of y's = number of x's)

$$\Rightarrow g(y_1, y_2, \cdots) dy_1 dy_2 \cdots = g(x_1, x_2, \cdots) \left| \frac{\partial(x_1, \cdots x_n)}{\partial(y_1, \cdots y_n)} \right| dx_1 dx_2 \cdots$$

$\left| \frac{\partial(x_1, \cdots x_n)}{\partial(y_1, \cdots y_n)} \right|$. is the Jacobian determinant of x's with respect to y's

Let us have two uniform deviates x_1, x_2 and the variables y_1, y_2 are defined as

$$y_1 = \sqrt{-2 \ln x_1} \cos 2\pi x_2$$
$$y_2 = \sqrt{-2 \ln x_1} \sin 2\pi x_2$$

Generalised Transformation Method

Equivalently

$$x_1 = \exp\left[-\frac{1}{2}(y_1^2 + y_2^2)\right]$$

$$x_2 = \frac{1}{2\pi}\tan^{-1}\left(\frac{y_2}{y_1}\right)$$

The Jacobian determinant:

$$\frac{\partial(x_1, x_2)}{\partial(y_1, y_2)} = \begin{vmatrix} \frac{\partial x_1}{\partial y_1} & \frac{\partial x_1}{\partial y_2} \\ \frac{\partial x_2}{\partial y_1} & \frac{\partial x_2}{\partial y_2} \end{vmatrix}$$

$$= -\left[\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{y_1^2}{2}\right)\right] \cdot \left[\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{y_2^2}{2}\right)\right]$$

Since this is a product of two functions, one of y_1 and one of y_2, one gets two independent random deviates following the Gaussian distribution of mean 0 and RMS 1:

$$g(y)dy = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{y^2}{2}\right)dy$$

Law of large numbers

- Concerns sum of a large number of random variables
- Choose n numbers x_i distributed randomly with uniform probability density in the interval a to b
- Evaluate $f(x_i)$ for each value of x_i

$$\frac{1}{n}\sum_{i=1}^{n} f(x_i) \quad \rightarrow \quad \frac{1}{b-a}\int_a^b f(x)dx \qquad \text{for large n}$$

The LHS is a consistent estimator of the integral which implies that the variance of f will be finite

The standard deviation of the estimator $\sim \sqrt{\frac{V(f)}{n}}$

This is the Central Limit theorem

The sum of a large number of independent random variables is always normally distributed, no matter how the random numbers are distributed

The normal distribution is specified by its expectation value a and variance σ^2:

Gaussian Distribution

$$g(x)dx = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right) dx$$

If u_i 's are uniform deviates between 0 and 1 $\rightarrow$ $g(u)du = du$

$$\langle u \rangle = \frac{\int_0^1 u \cdot du}{\int_0^1 du} = \frac{1}{2}$$

$$\text{variance } \sigma_u^2 = \frac{\int_0^1 (u - \frac{1}{2})^2 du}{\int_0^1 du} = \frac{1}{12}$$

So if one takes the sum of k variables: $u_{(k)} = \sum_{i=1}^{k} u_i$, then

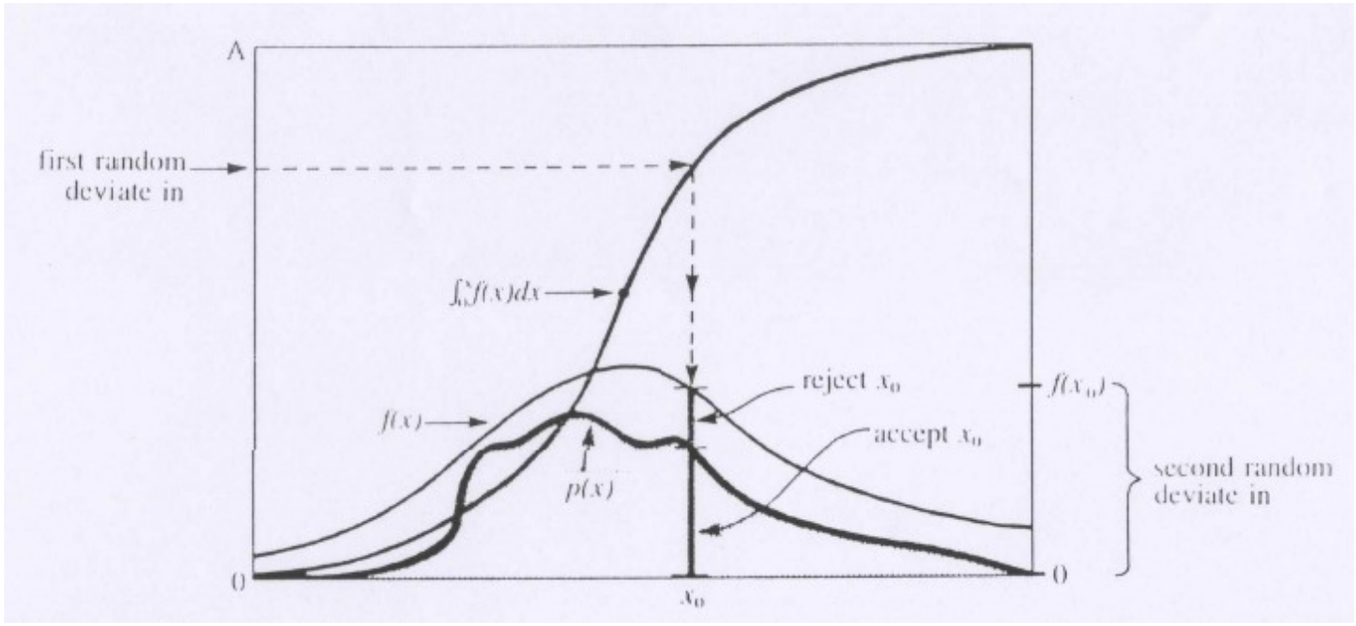
$$\langle u_{(k)} \rangle = \frac{k}{2}$$

$$\text{variance } \sigma_{u_{(k)}}^2 = \frac{k}{12}$$

By choosing 12 random variables and computing $\sum_{i=1}^{12} u_i - 6$ one gets a normally distributed variable with a mean 0 and variance 1

Rejection method

- Rejection method:
  If the probability distribution function is known and computable, this method can be applied. Here one does not require:
    Cumulative distribution function to be available
    The distribution function could be inverted



For example, one needs to generate a random number in the range a to b according to a probability distribution function proportional to p(x)
  choose a function f(x) (comparison function) such that
    The corresponding cumulative distribution function is computable and invertible
    f(x) lies above p(x) at all places between a and b
  choose x using the transformation method applied to the function f(x)
  use a second deviate u which is uniformly distributed between 0 and 1. If p(x)/f(x) ≤ u, then the value of x is to be accepted; otherwise that value of x is to be rejected and a fresh value has to be obtained using the transformation method

A more General method

- Express the probability density function as a sum of components:

$$F(x) = \sum_{i}^{n} \alpha_i \cdot f_i(x) \cdot g_i(x)$$

with α's positive; $f_i(x)$ normalised probability density functions which can be inverted; and $g_i(x)$ are computable functions which do satisfy the relation

$$0 \leq g_i(x) \leq 1.$$

Solution to the problem can be obtained as:

- select an integer i randomly within $0 \leq i \leq n$ and with the probability of selecting it: $i \propto \alpha_i$
- select a variable x' from $f_i(x)$ using the transformation method
- calculate $g_i(x)$ and accept x = x' with selection probability $g_i(x)$
- If rejected, go and select i once again and repeat

This is a good method of sampling x if

- all sub-distribution functions $f_i(x)$ can be easily sampled
- the rejection functions $g_i(x)$ can be quickly computed
- mean number of tries $(\sim \sum \alpha_i)$ is not too large

Application to Pair Production

- Pair production: $\gamma\gamma^* \rightarrow e^+e^-$

  A photon of energy E produces a pair of electron and positron with the electron carrying an energy fraction ϵ

$$\frac{d\sigma}{d\epsilon} = r_e^2 \frac{\alpha Z}{E^2} [Z + \xi(Z)] \left(\left[\epsilon^2 + (1-\epsilon)^2\right] [\phi_1(\delta) - F(Z)] + \frac{2}{3}\epsilon(1-\epsilon) [\phi_2(\delta) - F(Z)] \right)$$

$$\text{with } \delta = \text{screening variable} = \frac{136m}{Z^{\frac{1}{3}}E} \frac{1}{\epsilon(1-\epsilon)}$$

Rewrite the probability density function for

$$\begin{matrix} & & \alpha_i & \cdot & f_i & \cdot & g_i \end{matrix}$$

$$F(\epsilon) = \frac{[\frac{1}{2} - \frac{m}{E}]^2}{3} F_1(\delta_{min}) \cdot \frac{3}{[\frac{1}{2} - \frac{m}{E}]^3} (\epsilon - \frac{1}{2})^2 \cdot \frac{F_1(\delta)}{F_1(\delta_{min})}$$

$$+ \frac{1}{2} F_2(\delta_{min}) \cdot \frac{1}{[\frac{1}{2} - \frac{m}{E}]} \cdot \frac{F_2(\delta)}{F_2(\delta_{min})}$$

$$\text{with } F_1(\delta) = 3\phi_1(\delta) - \phi_2(\delta) - 2F(Z)$$

$$F_2(\delta) = \frac{3}{2}\phi_1(\delta) + \frac{1}{2}\phi_2(\delta) - 2F(Z)$$

$$\delta_{min} = \frac{136m}{Z^{\frac{1}{3}}E} \cdot 4$$

Pair Production

Kinematic range for $\epsilon \left(\frac{m}{E} \leq \epsilon \leq 1 - \frac{m}{E}\right)$

Using of symmetry of σ under $\epsilon \leftrightarrow 1 - \epsilon$ restrict $\left(\frac{m}{E} \leq \epsilon \leq \frac{1}{2}\right)$

Define a variable $B = \frac{\alpha_1}{\alpha_1 + \alpha_2}$

- Generate r_0 with $0 \leq r_0 \leq 1$. If $r_0 < B$, choose $i = 1$, else $i = 2$
- Generate r_1 with $0 \leq r_1 \leq 1$. If $i = 1$, $\epsilon = \frac{1}{2}\left[1 - \frac{m}{2E}(1 - r_1)^{\frac{1}{3}}\right]$.
  Otherwise estimate $\epsilon = \frac{m}{E} + \left(\frac{1}{2} - \frac{m}{E}\right) r_1$
- Generate r_2 with $0 \leq r_2 \leq 1$. If $r_2 \leq g_i(\epsilon)$, then accept this ϵ, else reject the current values of i, ϵ and try to find i again