12. Random Number Generation and Monte Carlo Method.

NUMERICAL ANALYSIS. Prof. Y. Nishidate (323-B, nisidate@u-aizu.ac.jp)
http://web-int.u-aizu.ac.jp/∼nisidate/na/

The Monte Carlo method is a unique and simple numerical method for solution of mathematical problems. The method "Monte Carlo" is named by a famous scientist von Neumann, and it is derived from the city Monte Carlo in Monaco famous for casinos. Actually, the method is based on **producing random numbers** and **do some trials** on the target function like gambles.

Random Number Generators

The Monte Carlo method requires a generation of random numbers. There are several ways to produce random numbers. One may be physical processes, or the other may be arithmetic processes. The former way may utilize the hardware noise (thermal or electromagnetic) which is naturally random, unpredictable, and unrepeatable. The pseudo-random number falls in the latter class which deterministically generate numbers. From a point of view, such as the encryption, pseudo-random numbers are often undesirable due to predictability and repeatability. However, for simulation purposes, the repeatability can be extremely useful.

Uniform Random Numbers

Linear Congruential Generator (LCG) The simplest type of pseudo-random random number is the Linear Congruential Generator (LCG). It produces the uniform probability distribution of numbers on the interval $0 \leq u \leq 1$, where u is a generated random number.

$$\begin{aligned} s_i &= a s_{i-1} + b \pmod{m}, \\ u_i &= s_i / m, \end{aligned}$$

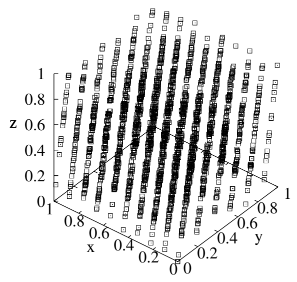
where (mod m) means modulo of m. s_i is the current internal state, which is less than m. As the starting point, it is necessary to select an integer $s_0 \neq 0$, called the **seed**. The random number generator of LCG type is widely used as standard algorithm, such as the rand function of C and java.util.Random.
For example, table below shows random numbers generated by the LCG with $a = 17$, $b = 0$, $m = 41$, $s_0 = 1$. It shows that the random generator has cycle. The cycle is too short, so that inappropriate for practical uses as a uniform random number generator.

i	s_i	u_i	i	s_i	u_i
1	17	0.414634	41	17	0.414634
2	2	0.048780	42	2	0.048780
3	34	0.829268	43	34	0.829268
4	4	0.097561	44	4	0.097561
	⋮				

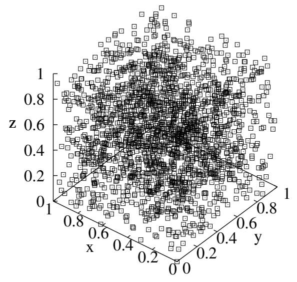
The $randu$ generator One of the variation of the LCG type random number generator is the $randu$ generator, used as standard algorithm on many early IBM computers and ported from there to many others. The $randu$ generator is obtained by using $a = 65539$, $b = 0$, $m = 2^{31}$ for the LCG

$$\begin{aligned} s_i &= 65539 s_{i-1} \pmod{2^{31}}, \\ u_i &= s_i / 2^{31}, \end{aligned}$$

However, the $randu$ generator is sometimes undesirable. The figure below shows the result of generating 2000 points within the three-dimensional unit cube. The random numbers generated by the $randu$ lie on 15 planes and insufficiently cover the unit cube.



$randu$



MSRNG

Minimal Standard Random Number Generator (MSRNG) Park and Miller proposed a simple LCG type generator with its parameters $a = 16807$, $b = 0$, $m = 2^{31} - 1$

$$\begin{aligned} s_i &= 16807 s_{i-1} \pmod{2^{31} - 1}, \\ u_i &= s_i / (2^{31} - 1), \end{aligned}$$

This is called the Minimal Standard Random Number Generator (MSRNG). The result of generating 2000 number of points within the unit cube is shown in the figure above. In contrast to the $randu$, the MSRNG demonstrates that it uniformly covers the unit cube.

Random Numbers in Arbitrary Range and Dimension

The generators introduced here produces uniform random numbers within the range $[0, 1]$. It is possible to map them in an interval $[a, b]$ using simple linear transformation.

$$v = (b - a)u + a,$$

where u is the random number in the interval $[0, 1]$ and v is the random number mapped in an interval $[a, b]$. Also, random numbers in D-dimensions can be realized by simply calling the random number generator D times.

Monte Carlo Method

Type 1: Approximation by Average

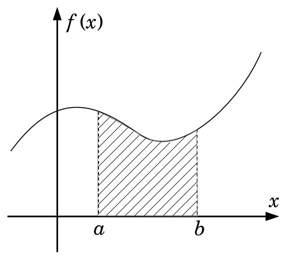
Consider the integration of the function $f(x)$ over the range $a \leq x \leq b$, which is also interpreted as summation of infinite number of small area fragments within the range.

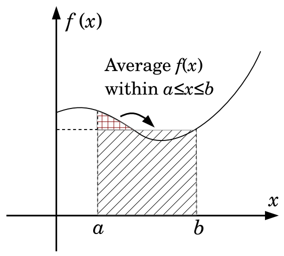
$$I = \int_a^b f(x)dx = \lim_{n \to \infty} \frac{b - a}{n} \sum_i^n f(\xi_i)$$

where $a \leq \xi_i \leq b$. For example, the numerical integrations derived from the Newton-Cotes formula use equally spaced sampling points, i.e. $\xi_i = a + i\frac{b-a}{n}$. Using a random value $a \leq u_i \leq b$ instead of ξ_i yields the Monte Carlo method for numerical integration

$$I \approx I_n = \frac{b - a}{n} \sum_i^n f(u_i).$$

This means that the integral is approximated by average of function value over the range $a \leq x \leq b$.





Type 2: Approximation by Probability

Find the "N-dimensional volume" which satisfy the following relations

$$\begin{aligned} f_1(\mathbf{x}) &< 0, \\ f_2(\mathbf{x}) &< 0, \\ &\vdots \\ f_M(\mathbf{x}) &< 0, \end{aligned}$$

where $\mathbf{x} = \{x^{(1)}, x^{(2)}, \cdots, x^{(N)}\}$ (N-dimensional vector) and M is the number of relations. For example, in two-dimensional case $\mathbf{x} = \{x^{(1)}, x^{(2)}\} = \{x, y\}$, and thus the "$N$-dimensional volume" is an area, etc.

We surround the given functions f_i in a N-dimensional bounding box, and evaluate the function values at points $\mathbf{x}_k$ generated by a random number generator. Then, count if $\mathbf{x}_k$ satisfies all relations $f_i < 0$. It is easy to evaluate whether the point $\mathbf{x}_k$ satisfies $f_i < 0$. The procedure may be summarized as follows

1. Define a rectangular N dimensional bounding box to cover the region of interest. Let's denote the bounding box by $x^{(j)}_{\min} \le x^{(j)} \le x^{(j)}_{\max}$ where $x^{(j)}$ is the coordinate along jth axis (The bounding box is N dimensional hypercube).

2. Produce n points in N-dimensional space within $x^{(j)}_{\min} \le x^{(j)} \le x^{(j)}_{\max}$ by a random number generator.

3. Count the number of points m which satisfies $f_i(\mathbf{x}^{(k)}) < 0$ where components of $\mathbf{x}^{(k)}$ are the random numbers generated in step 2.

Now, we have m points which satisfies $f_i < 0$ among total of n trials. Therefore, the probability of satisfying the conditions is approximated as m/n. We covered the range $x^{(j)}_{\min} \le x^{(j)} \le x^{(j)}_{\max}$ by the random number, so its "N-dimensional volume" V is multiplication of length along all axes, i.e.,

$$\begin{aligned} V &= \prod_i^N (x^{(i)}_{\max} - x^{(i)}_{\min}) \\ &= (x^{(1)}_{\max} - x^{(1)}_{\min}) \times (x^{(2)}_{\max} - x^{(2)}_{\min}) \times \cdots \times (x^{(N)}_{\max} - x^{(N)}_{\min}). \end{aligned}$$

Therefore, the volume inside the set of relations $f_i < 0$ is approximated as the expected value

$$I \approx I_n = V \frac{m}{n}.$$

This idea can be useful if it is difficult or impossible to find the "volume" using standard numerical integration algorithms, e.g. the trapezoid, Simpson's rules, etc. Also, the standard numerical algorithms are sometimes less evident on extending the idea to multi-dimensional integrations and requires n^D sampling points where D is the dimension of the integration. Thus, the Gauss quadrature rule or the Monte Carlo integration is widely used in many practical applications.

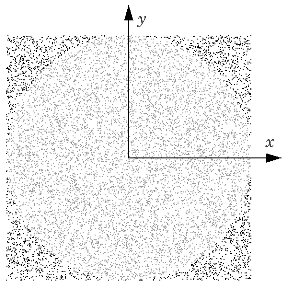
Example: Let's consider the following simple example

$$f_1(x, y) = x^2 + y^2 - 2^2 < 0.$$

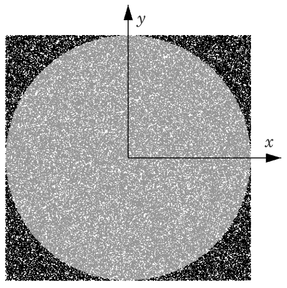
It is evident that f_1 expresses the inside of circle of radius 2, and therefore its area is 4π. There is only one condition $f_1 < 0$, so we produce n random points in 2D by a random number generator, suppose the MSRNG with $s_0 = 1$. Region of interest is $-2 \le x, y \le 2$, so that the uniform random points in $0 \le u_i \le 1$ should be scaled by factor of 4 and translated by -2. The 2D volume of bounding box is the area of rectangular region, 16. Therefore, the area inside f_1 is approximated as the expected value of area

$$I \approx I_n = 16 \frac{m}{n}.$$

The results of simulation with $n = 10000$ and $n = 100000$ are shown in the figures below, where points satisfy $f_1 < 0$ are gray and are black otherwise.



$n = 10000$, $I_n = 3.974\pi$



$n = 100000$, $I_n = 4.001\pi$

Convergence of Monte Carlo Method

An important question arises that how much the error may decrease as the number of trial n increases. It is possible to estimate the convergence rate in a statistical sense. Let's denote the random variable by X and its probability distribution by $p(X)$. We want to integrate function $f(x)$, and the expected value is

$$E[f(X)] = \int_{-\infty}^{\infty} f(X)p(X)dX,$$

which corresponds to the exact solution. On the other hand, we want to estimate the exact solution by the following finite sum

$$E'[f(X)] = \frac{1}{n} \sum_i^n f(X_i),$$

where $E'[f(X)]$ is an estimation of $E[f(X)]$. It is assured by the law of large numbers of the probability theory that the average of large number of trials tend to converge the expected value. Therefore, it is expected that the results of Monte Carlo simulation become more precise as number of trial increases. The error of Monte Carlo simulation can be written as

$$E'[f(X)] - E[f(X)] = \frac{1}{n} \sum_i^n f(X_i) - E[f(X)].$$

According to the central limit theorem of the probability theory, the error distribution converges to the normal distribution with its average 0, variance σ^2/n, and standard deviation $\sigma/\sqrt{n}$, where

$$\sigma^2 = \int_{-\infty}^{\infty} f^2(X)p(X)dX - \{E[f(X)]\}^2.$$

The estimation $E'[f(X)]$ becomes within the deviation 3 multiplied by the standard deviation $\sigma/\sqrt{n}$, with the probability of 99% due to the characteristics of the normal distribution. In other words, $E'[f(X)]$ satisfies the following relation in 99%.

$$E'[f(X)] = E[f(X)] \pm \frac{3\sigma}{\sqrt{n}}$$

This is an estimation of convergence order $O(n^{-1/2})$ of Monte Carlo simulation. Note that the precision also depends on σ, that is, the probability distribution function $p(X)$. This implies that characteristics of random number may contribute to the convergence of the Monte Carlo method.