3
The Monte Carlo method

The Monte Carlo method is a numerical technique for calculating probabilities and related quantities by using sequences of random numbers. For the case of a single random variable, the procedure can be broken into the following stages. First, a series of random values $r_1, r_2 \ldots$ is generated according to a uniform distribution in the interval $0 < r < 1$. That is, the p.d.f. $g(r)$ is given by

$$g(r) = \begin{cases} 1 & 0 < r < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Next, the sequence $r_1, r_2, \ldots$ is used to determine another sequence $x_1, x_2 \ldots$ such that the x values are distributed according to a p.d.f. $f(x)$ in which one is interested. The values of x can then be treated as simulated measurements, and from them the probabilities for x to take on values in a certain region can be estimated. In this way one effectively computes an integral of $f(x)$. This may seem like a trivial exercise, since the function $f(x)$ was available to begin with, and could simply have been integrated over the region of interest. The true usefulness of the technique, however, becomes apparent in multidimensional problems, where integration of a joint p.d.f. $f(x, y, z, \ldots)$ over a complicated region may not be feasible by other methods.

3.1 Uniformly distributed random numbers

In order to generate a sequence of uniformly distributed random numbers, one could in principle make use of a random physical process such as the repeated tossing of a coin. In practice, however, this task is almost always accomplished by a computer algorithm called a **random number generator**. Many such algorithms have been implemented as user-callable subprograms (e.g. the routines `RANMAR` [Mar91] or `RANLUX` [Lüs94, Jam94], both in [CER97]). A detailed discussion of random number generators is beyond the scope of this book and the interested reader is referred to the more complete treatments in [Bra92, Jam90]. Here a simple but effective algorithm will be presented in order to illustrate the general idea.

A commonly used type of random number generator is based on the **multiplicative linear congruential** algorithm. Starting from an initial integer value n_0 (called the **seed**), one generates a sequence of integers $n_1, n_2, \ldots$ according to the rule

$$n_{i+1} = (an_i) \bmod m. \tag{3.2}$$

Here the **multiplier** a and **modulus** m are integer constants and the mod (modulo) operator means that one takes the remainder of an_i divided by m. The values n_i follow a periodic sequence in the range $[1, m-1]$. In order to obtain values uniformly distributed in $(0, 1)$, one uses the transformation

$$r_i = n_i/m. \tag{3.3}$$

Note that this excludes 0 and 1; in some other algorithms these values can be included. The initial value n_0 and the two constants a and m determine the entire sequence, which, of course, is not truly random, but rather strictly determined. The resulting values are therefore more correctly called **pseudorandom**. For essentially all applications these can be treated as equivalent to true random numbers, with the exception of being reproducible, e.g. if one repeats the procedure with the same seed.

The values of m and a are chosen such that the generated numbers perform well with respect to various tests of randomness. Most important among these is a long period before the sequence repeats, since after this occurs the numbers can clearly no longer be regarded as random. In addition, one tries to attain the smallest possible correlations between pairs of generated numbers. For a 32-bit integer representation, for example, $m = 2147483399$ and $a = 40692$ have been shown to give good results, and with these one attains the maximum period of $m - 1 \approx 2 \times 10^9$ [Lec88]. More sophisticated algorithms allow for much longer periods, e.g. approximately 10^{43} for the `RANMAR` generator [Mar91, CER97].

3.2 The transformation method

Given a sequence of random numbers $r_1, r_2, \ldots$ uniformly distributed in $[0, 1]$, the next step is to determine a sequence $x_1, x_2, \ldots$ distributed according to the p.d.f. $f(x)$ in which one is interested. In the transformation method this is accomplished by finding a suitable function $x(r)$ which directly yields the desired sequence when evaluated with the uniformly generated r values. The problem is clearly related to the transformation of variables discussed in Section 1.4. There, an original p.d.f. $f(x)$ for a random variable x and a function $a(x)$ were specified, and the p.d.f. $g(a)$ for the function a was then found. Here the task is to find a function $x(r)$ that is distributed according to a specified $f(x)$, given that r follows a uniform distribution between 0 and 1.

The probability to obtain a value of r in the interval $[r, r+dr]$ is $g(r)dr$, and this should be equal to the probability to obtain a value of x in the corresponding interval $[x(r), x(r)+dx(r)]$, which is $f(x)dx$. In order to determine $x(r)$ such that this is true, one can require that the probability that r is less than some value r' be equal to the probability that x is less than $x(r')$. (We will see in the following example that this prescription is not unique.) That is, one must find a function $x(r)$ such that $F(x(r)) = G(r)$, where F and G are the cumulative distributions corresponding to the p.d.f.s f and g. Since the cumulative distribution for the uniform p.d.f. is $G(r) = r$ with $0 \leq r \leq 1$, one has

$$\begin{aligned} F(x(r)) = \int_{-\infty}^{x(r)} f(x')dx' &= \int_{-\infty}^{r} g(r')dr' \\ &= r. \end{aligned} \tag{3.4}$$

Equation (3.4) says in effect that the cumulative distribution $F(x)$, treated as a random variable, is uniformly distributed between 0 and 1 (cf. equation (2.18)).

Depending on the $f(x)$ in question, it may or may not be possible to solve for $x(r)$ using equation (3.4). Consider the exponential distribution discussed in Section 2.4. Equation (3.4) becomes

$$\int_{0}^{x(r)} \frac{1}{\xi} e^{-x'/\xi} dx' = r. \tag{3.5}$$

Integrating and solving for x gives

$$x(r) = -\xi \log(1 - r). \tag{3.6}$$

If the variable r is uniformly distributed between 0 and 1 then $r' = 1 - r$ clearly is too, so that the function

$$x(r) = -\xi \log r \tag{3.7}$$

also has the desired property. That is, if r follows a uniform distribution between 0 and 1, then $x(r) = -\xi \log r$ will follow an exponential distribution with mean ξ.

3.3 The acceptance–rejection method

It turns out to be too difficult in many practical applications to solve equation (3.4) for $x(r)$ analytically. A useful alternative is von Neumann's acceptance–rejection technique [Neu51]. Consider a p.d.f. $f(x)$ which can be completely surrounded by a box between x_{min} and x_{max} and having height f_{max}, as shown in Fig. 3.1. One can generate a series of numbers distributed according to $f(x)$ with the following algorithm:

(1) Generate a random number x, uniformly distributed between x_{min} and x_{max}, i.e. $x = x_{\text{min}} + r_1(x_{\text{max}} - x_{\text{min}})$ where r_1 is uniformly distributed between 0 and 1.

(2) Generate a second independent random number u uniformly distributed between 0 and f_{max}, i.e. $u = r_2 f_{\text{max}}$.

(3) If $u < f(x)$, then accept x. If not, reject x and repeat.

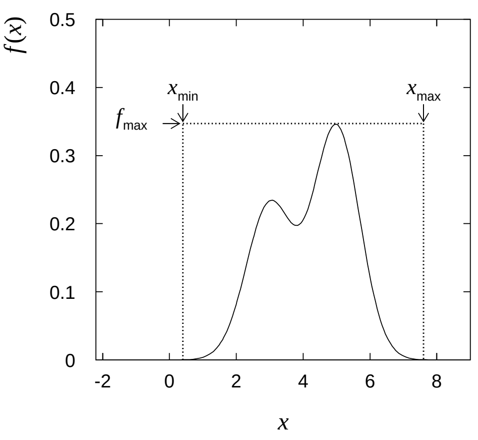

Fig. 3.1 A probability density $f(x)$ enclosed by a box to generate random numbers using the acceptance-rejection technique.

The accepted x values will be distributed according to $f(x)$, since for each value of x obtained from step (1) above, the probability to be accepted is proportional to $f(x)$.

As an example consider the p.d.f.[1]

$$f(x) = \frac{3}{8}\,(1 + x^2), \quad -1 \le x \le 1. \tag{3.8}$$

At $x = \pm 1$ the p.d.f. has a maximum value of $f_{\max} = 3/4$. Figure 3.2(a) shows a scatter plot of the random numbers u and x generated according to the algorithm given above. The x values of the points that lie below the curve are accepted. Figure 3.2(b) shows a normalized histogram constructed from the accepted points.

The efficiency of the algorithm (i.e. the fraction of x values accepted) is the ratio of the areas of the p.d.f. (unity) to that of the enclosing box $f_{\max} \cdot (x_{\max} - x_{\min})$. For a highly peaked density function the efficiency may be quite low, and the algorithm may be too slow to be practical. In cases such as these, one can improve the efficiency by enclosing the p.d.f. $f(x)$ in any other curve $g(x)$ for which random numbers can be generated according to $g(x)/\int g(x')dx'$, using, for example, the transformation method.

The more general algorithm is then:

(1) Generate a random number x according to the p.d.f. $g(x)/\int g(x')dx'$.
(2) Generate a second random number u uniformly distributed between 0 and $g(x)$.
(3) If $u < f(x)$, then accept x. If not, reject x and repeat.

[1]Equation (3.8) gives the distribution of the scattering angle θ in the reaction $e^+ e^- \to \mu^+ \mu^-$ with $x = \cos\theta$ (see e.g. [Per87]).

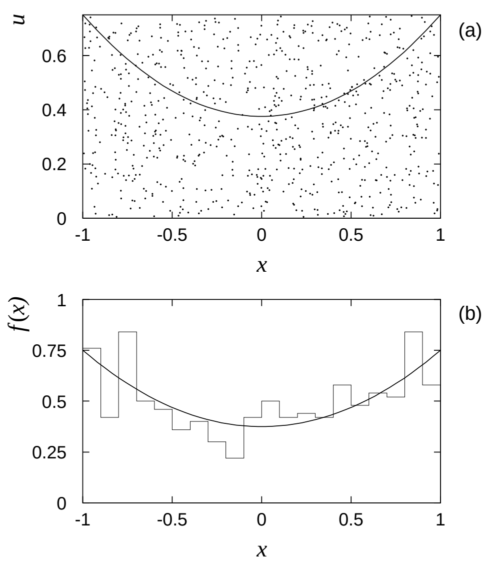

Fig. 3.2 (a) Scatter plot of pairs of numbers (u, x), where x is uniformly distributed in $-1 \leq x \leq 1$, and u is uniform in $0 \leq u \leq f_{\max}$. The x values of the points below the curve are accepted. (b) Normalized histogram of the accepted x values with the corresponding p.d.f.

Here the probability to generate a value x in step (1) is proportional to $g(x)$, and the probability to be retained after step (3) is equal to $f(x)/g(x)$, so that the total probability to obtain x is proportional to $f(x)$ as required.

3.4 Applications of the Monte Carlo method

The Monte Carlo method can be applied whenever the solution to a problem can be related to a parameter of a probability distribution. This could be either an explicit parameter in a p.d.f., or the integral of the distribution over some region. A sequence of Monte Carlo generated values is used to evaluate an estimator for the parameter (or integral), just as is done with real data. Techniques for constructing estimators are discussed in Chapters 5–8.

An important feature of properly constructed estimators is that their statistical accuracy improves as the number of values n in the data sample (from Monte Carlo or otherwise) increases. One can show that under fairly general conditions, the standard deviation of an estimator is inversely proportional to $\sqrt{n}$ (see Section 6.6). The Monte Carlo method thus represents a numerical integration technique for which the accuracy increases as $1/\sqrt{n}$.

This scaling behavior with the number of generated values can be compared to the number of points necessary to compute an integral using the trapezoidal rule. Here the accuracy improves as $1/n^2$, i.e. much faster than by Monte Carlo. For an integral of dimension d, however, this is changed to $1/n^{2/d}$, whereas for Monte Carlo integration one has $1/\sqrt{n}$ for any dimension. So for $d > 4$, the dependence of the accuracy on n is better for the Monte Carlo method. For other integration methods, such as Gaussian quadrature, a somewhat better rate of convergence can be achieved than for the trapezoidal rule. For a large enough number of dimensions, however, the Monte Carlo method will always be superior. A more detailed discussion of these considerations can be found in [Jam80].

The Monte Carlo technique provides a method for determining the p.d.f.s of functions of random variables. Suppose, for example, one has n independent random variables $x_1, \ldots, x_n$ distributed according to known p.d.f.s $f_1(x_1)$, ..., $f_n(x_n)$, and one would like to compute the p.d.f. $g(a)$ of some (possibly complicated) function $a(x_1, \ldots, x_n)$. The techniques described in Section 1.4 are often only usable for relatively simple functions of a small number of variables. With the Monte Carlo method, a value for each x_i is generated according to the corresponding $f_i(x_i)$. The value of $a(x_1, \ldots, x_n)$ is then computed and recorded (e.g. in a histogram). The procedure is repeated until one has enough values of a to estimate the properties of its p.d.f. $g(a)$ (e.g. mean, variance) with the desired statistical precision. Examples of this technique will be used in the following chapters.

The Monte Carlo method is often used to simulate experimental data. In particle physics, for example, this is typically done in two stages: event generation and detector simulation. Consider, for example, an experiment in which an incoming particle such as an electron scatters off a target and is then detected. Suppose there exists a theory that predicts the probability for an event to occur as a function of the scattering angle (i.e. the differential cross section). First one constructs a Monte Carlo program to generate values of the scattering angles, and thus the momentum vectors, of the final state particles. Such a program is called an **event generator**. In high energy physics, event generators are available to describe a wide variety of particle reactions.

The output of the event generator, i.e. the momentum vectors of the generated particles, is then used as input for a **detector simulation program**. Since the response of a detector to the passage of the scattered particles also involves random processes such as the production of ionization, multiple Coulomb scattering, etc., the detector simulation program is also implemented using the Monte Carlo method. Programming packages such as GEANT [CER97] can be used to describe complicated detector configurations, and experimental collaborations typically spend considerable effort in achieving as complete a modeling of the detector as possible. This is especially important in order to optimize the detector's design for investigating certain physical processes before investing time and money in constructing the apparatus.

When the Monte Carlo method is used to simulate experimental data, one can most easily think of the procedure as a computer implementation of an intrinsically random process. Probabilities are naturally interpreted as relative frequencies of outcomes of a repeatable experiment, and the experiment is simply repeated many times on the computer. The Monte Carlo method can also be regarded, however, as providing a numerical solution to any problem that can be related to probabilities. The results are clearly independent of the probability interpretation. This is the case, for example, when the Monte Carlo method is used simply to carry out a transformation of variables or to compute integrals of functions which may not normally be interpreted as probability densities.