1

Simulations and the Monte Carlo method

Introduction

In order to introduce the Monte Carlo method, let us consider a problem of numerical integration. There exist several numerical methods for the approximate computation of the integral

$$\int_{[0,1]} f(x)dx,$$

based on formulae of the type $\sum_{i=1}^{n} w_i f(x_i)$, where the w_i are positive numbers whose sum equals 1 and the x_i are points in the interval $[0, 1]$. For example, if $w_i = 1/n$, $1 \leq i \leq n$, and $x_i = i/n$, this is the trapezoid rule. But there exist other approximations, such as Simpson's rule and the Gaussian quadrature formula. A Monte Carlo method is of the same type: we choose $w_i = 1/n$, and we choose the x_i 'at random' (meaning here according to the uniform law on $[0, 1]$, later denoted by $\mathcal{U}(0, 1)$). As we shall see below, the convergence is guaranteed by the law of large numbers, and the rate of convergence, of order $n^{-1/2}$, is given by the central limit theorem. Clearly, that rate of convergence may seem rather slow, if we compare it with the rate of other numerical integration methods in dimension 1. But all these numerical methods collapse if we go to higher dimensions. Indeed, in all these methods, the precision is a function of the distance between two contiguous points of the discretization. But if we use n points for the discretization of $[0, 1]^d$, the distance between two contiguous points is of order $n^{-1/d}$, hence if we want a precision of order $1/n$ with a 'first-order' method of approximation of an integral over $[0, 1]^d$, the number of points we need is of order n^d. On the other hand, the Monte Carlo method is essentially unaffected by the dimension.

Historically, the method goes back to Count Buffon who described in 1777 a method for the approximate computation of π, based on the realization of repeated

Markov Processes and Applications: Algorithms, Networks, Genome and Finance E. Pardoux

experiments. But the true birth of the Monte Carlo method is linked to the appearance of the first computers. The first papers describing methods of this type date back from the late 1940s and early 1950s. These methods continue to grow more and more popular. This is in large part due to the simplicity with which one can program them, as well as the ability of today's computers to perform a huge number of random draws in a reasonable length of time.

1.1 Description of the method

If we wish to use a Monte Carlo method, we need first to write the quantity of interest as the expectation of a random variable. This is often easy, as in the case of the computation of an integral, but it might be much more involved, as when we wish to solve a parabolic or elliptic partial differential equation (see Sections 7.9 and 9.3 below).

The next step is to compute a quantity of the form $\mathbb{E}(X)$, where X is a random variable. In order do so, we need to be able to simulate mutually independent random variables $X_1, \ldots, X_n$, all having the law of X. It then remains to approximate $\mathbb{E}(X)$ by

$$\mathbb{E}(X) \approx \frac{1}{n}(X_1 + \ldots + X_n).$$

Let us describe one example of the application of the Monte Carlo method, to the computation of an integral. We will explain in detail the two steps presented above: how to write the integral as an expectation, and how to simulate the random variables. Suppose that we wish to compute an integral of the form

$$I = \int_{[0,1]^d} f(u_1, \ldots, u_d) du_1 \ldots du_d.$$

We set $X = f(U_1, \ldots, U_d)$, where the U_i, $i = 1, \ldots, d$, are independent and identically distributed (i.i.d.) random variables, each one having the law $\mathcal{U}(0, 1)$. We have

$$\mathbb{E}(X) = \mathbb{E}\left(f(U_1, \ldots, U_d)\right) = \int_{[0,1]^d} f(u_1, \ldots, u_d) du_1 \ldots du_d.$$

We have just completed the first step – our integral is written as an expectation.

For the simulation, suppose we can produce a sequence $(U_i, i \geq 1)$ of i.i.d. random variables whose common law is $\mathcal{U}(0, 1)$. We define $X_1 = f(U_1, \ldots, U_d)$, $X_2 = f(U_{d+1}, \ldots, U_{2d})$, etc. Then the sequence $(X_i, i \geq 1)$ is an i.i.d. sequence of random variables, all having the same law as X. We can now implement the Monte Carlo method.

It is important to note the simplicity with which the corresponding program can be written. Note also that no specific regularity of f is required. f need only be integrable.

One often needs to compute a more general type of integral, namely

$$I = \int_{\mathbb{R}^d} g(x) f(x) dx = \int_{\mathbb{R}^d} g(x_1, \ldots, x_d) f(x_1, \ldots, x_d) dx_1 \ldots dx_d,$$

with $f(x)$ non-negative and $\int f(x)dx = 1$. Then I equals $\mathbb{E}(g(X))$ if X is an $\mathbb{R}^d$-valued random variable whose law is $f(x)dx$. The problem now is to simulate random vectors having that probability law. Some answers, related to commonly used probability laws, will be given in Section 1.3 below.

But let us first answer the two questions:

- When and why does this algorithm converge?
- Can we get a precise idea of the accuracy of this algorithm?

1.2 Convergence theorems

The answers to the two above questions are given by the two most fundamental theorems in the calculus of probability, namely the law of large numbers, which permits us to establish the convergence of the method, and the central limit theorem, which gives a precise indication of its rate of convergence.

Theorem 2.1 *Let $(X_n, n \geq 1)$ be a sequence of i.i.d. random variables, all having the law of X. If $\mathbb{E}(|X|) < +\infty$, then, for $\mathbb{P}$ almost all ω (this means that there exists $N \subset \Omega$, with $\mathbb{P}(N) = 0$ and such that whenever $\omega \notin N$),*

$$\mathbb{E}(X) = \lim_{n \to +\infty} \frac{1}{n}(X_1 + \ldots + X_n)(\omega).$$

The evaluation of the method relies upon estimating the error

$$\varepsilon_n = \mathbb{E}(X) - \frac{1}{n}(X_1 + \ldots + X_n).$$

The central limit theorem gives the asymptotic behaviour of the quantity ε_n, which has a random nature. It says that the law of ε_n tends to look like a centred Gaussian law.

Theorem 2.2 *Let $(X_n, n \geq 1)$ be a sequence of i.i.d. random variables, all having the law of X. Assume that $\mathbb{E}(X^2) < +\infty$. Let σ^2 denote the variance of X:*

$$\sigma^2 = \mathbb{E}(X^2) - \mathbb{E}(X)^2 = \mathbb{E}\left((X - \mathbb{E}(X))^2\right).$$

Then

$$\frac{\sqrt{n}}{\sigma}\varepsilon_n \text{ converges in law towards } Z \simeq N(0, 1).$$

In other words, for all $a < b$,

$$\lim_{n \to +\infty} \mathbb{P}\left(\frac{\sigma}{\sqrt{n}}a \leq \varepsilon_n \leq \frac{\sigma}{\sqrt{n}}b\right) = \int_a^b e^{-x^2/2}\frac{dx}{\sqrt{2\pi}}.$$

In practice, if n is not too small (which will always be the case in the situation of a Monte Carlo computation), the above probability can be replaced by its limit, hence we may act as if ε_n were a centred Gaussian random variable with variance σ^2/n.

Remark 2.3 *1. This result is extremely powerful, since it gives us a rate of convergence which can be easily estimated with the help of the simulations which have already been realized. The fact that we have a reliable estimate of the error, without any further computation, is a real strength of the method.*

2. However, the central limit theorem never provides a bound for the error, since the support of a Gaussian random variable is $\mathbb{R}$*. One way to describe the error in the Monte Carlo method is either by providing the standard deviation of* ε_n*, which is equal to* $\sigma/\sqrt{n}$*, or else by providing a* 95% *confidence interval for the result. This means that there is a* 0.95 *chance that the quantity of interest is in the given interval (and hence there is a* 0.05 *chance that it is outside that interval). Clearly* 0.95 *can be replaced buy any value close to* 1.

Note the important role played by the variance of X in the estimation of the error. Since we can choose the law of X, with the restriction that $\mathbb{E}(X)$ be the quantity which we are interested in, we may wish to replace X by another random variable with the same expectation and a smaller variance. Such a procedure is called a variance reduction method (see Section 1.4 below).

We should also note that the rate at which the error goes to 0 is not very fast. However, there are several situations where this slowly converging method is the only available one (e.g. integral or parabolic partial differential equations in dimension higher than 4). It is also remarkable that the rate of convergence does not depend upon the smoothness of the function f.

We will now describe the use of the central limit theorem for analysing the rate of convergence of the Monte Carlo method, in two examples. This will allow us to present a limitation of the use of the Monte Carlo method.

A good case Suppose we wish to compute $p = \mathbb{P}(Y \leq \lambda)$, where Y is a random variable with an arbitrary law. Define $X = \mathbf{1}_{\{Y \leq \lambda\}}$. Then $\mathbb{E}(X) = p$, and $\sigma^2 = \mathrm{var}(X) = p(1-p)$. Consequently, after n independent draws $X_1, \ldots, X_n$ of X, we have

$$p_n = \frac{X_1 + \ldots + X_n}{n} \approx p + \frac{\sigma}{\sqrt{n}} Z.$$

Since $p(1-p) \leq 1/4$, if we want the standard deviation $\sigma/\sqrt{n}$ of the error to be of the order of 0.01, we should choose n of the order of 2500. If we choose $n = 2500$, the 0.95 confidence interval for p is then, according to the central limit theorem, $[p_n - 1.96 \times 0.01, p_n + 1.96 \times 0.01]$. If the true unknown value p is of the order of 0.50, this leads to an acceptable error.

However, if the true value of p is very small, the above value of n may be insufficient, if we want the error to be smaller than the quantity to be estimated. We need a number of simulations of the order of $1/p$.

A tough case Imagine that we wish to compute $\mathbb{E}(\exp(\beta Z))$, where Z is an $N(0, 1)$ random variable. Clearly

$$E = \mathbb{E}\left(e^{\beta Z}\right) = e^{\beta^2/2}.$$

If we apply a Monte Carlo method to this case, we let $X = e^{\beta Z}$. The variance of X is $\sigma^2 = e^{2\beta^2} - e^{\beta^2}$. After n simulations $X_1, \ldots, X_n$ according to the law of X, we have

$$E_n = \frac{X_1 + \ldots + X_n}{n} \approx E + \frac{\sigma}{\sqrt{n}} Z.$$

The standard deviation of the relative error is

$$\frac{\sigma}{E\sqrt{n}} = \sqrt{\frac{e^{\beta^2} - 1}{n}}.$$

If we want that quantity to be smaller than a given $\varepsilon > 0$, then we should choose $n \approx \varepsilon^{-2}(e^{\beta^2} - 1)$. If $\varepsilon = 1$ and $\beta = 5$, this means $n = 7 \times 10^{10}$, which is far too high. After 10^5 simulations, the 0.95 confidence interval might be $[-467\,647, 2\,176\,181]$, which is a disaster. The only positive point is that we are aware of the fact that our estimate is terrible, at least if we have a good estimate of the variance of the X_n.

This example shows a practical limitation of the Monte Carlo method, when we use random variables with large variances. This leads us to formulate the following rule: in any Monte Carlo computation, one must exploit the simulations, in order to estimate the variance of the random variable whose expectation we wish to compute.

Note that reducing the variance of the random variable to be simulated is often a crucial step in making a Monte Carlo computation efficient. We shall discuss this issue in Section 1.4.

1.3 Simulation of random variables

Simulation of $\mathcal{U}(0, 1)$ Any programming language today possesses a *pseudo-random* number generator. Such a program produces as output a perfectly deterministic (and also periodic) sequence, but whose statistical properties resemble those of a sequence of independent realizations of the law $\mathcal{U}(0, 1)$. The problem of inventing a good 'random number generator' is to create a recurrence formula which, in a reasonable time, produces a sequence of numbers which looks as much as possible like a sequence of realizations of independent $\mathcal{U}(0, 1)$ random variables, with a period which should be as large as possible. The study of those generators is part of the theory of dynamical systems. Most classical algorithms generating pseudo-random numbers are presented in [23] and [32], among others. More recently, Matsumoto and Nishimura [26] proposed a generator with period $2^{19\,937} - 1$!

Note that all random number generators try in fact to deliver draws from a uniform law on $\{1/M, 2/M, \ldots, (M-1)/M, 1\}$, with M very, very large.

It remains to simulate laws other than the uniform law.

Simulation of a Bernoulli random variable Let $0 < p < 1$. If U is a $\mathcal{U}(0, 1)$ random variable, $X = \mathbf{1}_{\{U \leq p\}}$ is a Bernoulli random variable with parameter p.

Simulation of a binomial random variable If $U_1, \ldots, U_n$ are independent $\mathcal{U}(0, 1)$ random variables, then

$$X = \mathbf{1}_{\{U_1 \leq p\}} + \ldots + \mathbf{1}_{\{U_n \leq p\}}$$

is a $B(n, p)$ random variable (binomial with parameters n and p).

Simulation of a geometric random variable $X = \inf\{k \geq 1;\ U_k \leq p\}$ is a geometric random variable with parameter p. A more efficient simulation procedure, based on the next lemma, is proposed in Exercise 5.1.

Inversion of the distribution function Recall the following classical result:

Lemma 3.1 *Let X be a random variable, and F its distribution function (i.e. $F(x) = \mathbb{P}(X \leq x)$). Define, for $0 \leq t \leq 1$,*

$$F^{-1}(t) = \inf\{x;\ F(x) > t\}.$$

Then if U has the law $\mathcal{U}[0, 1]$, $F^{-1}(U)$ has the same law as X.

PROOF This is immediate:

$$\mathbb{P}(F^{-1}(U) \leq x) = \mathbb{P}(U \leq F(x)) = F(x).$$

Indeed, $\{t; F^{-1}(t) \leq x\} \subset \{t; t \leq F(x)\}$, and the difference between those two sets is at most a one point set.

□

This method can be used whenever we have an explicit expression for the inverse of F. This is particularly the case for the exponential probability law.

Simulation of an exponential random variable Recall that a random variable X has the exponential law with parameter λ whenever, for all $t \in \mathbb{R}_+$,

$$\mathbb{P}(X > t) = \exp(-\lambda t).$$

Hence, if F is the distribution function of X, $F(t) = 1 - e^{-\lambda t}$, and

$$F^{-1}(x) = -\frac{\log(1-x)}{\lambda}.$$

If $U \simeq \mathcal{U}[0, 1]$, the same is true with $1 - U$, and

$$-\frac{\log U}{\lambda} \simeq \mathcal{E}(\lambda).$$

Simulation of Gaussian random variables (Box–Müller algorithm) A classical method for the simulation of Gaussian random variables is based on the remark that, if U and V are two independent $\mathcal{U}(0, 1)$ random variables,

$$\sqrt{-2\log(U)}\cos(2\pi V) \quad \text{and} \quad \sqrt{-2\log(U)}\sin(2\pi V)$$

are independent $N(0, 1)$ random variables. One can check this result as follows. If X and Y are independent $N(0, 1)$ random variables, $f : \mathbb{R}^2 \to \mathbb{R}_+$,

$$\begin{aligned}
\mathbb{E}f(X, Y) &= \frac{1}{2\pi}\int_{\mathbb{R}}\int_{\mathbb{R}} \exp\left(-\frac{x^2+y^2}{2}\right) f(x, y)dxdy \\
&= \frac{1}{2\pi}\int_0^{2\pi}\int_0^{\infty} r\exp\left(-\frac{r^2}{2}\right) f(r\cos\theta, r\sin\theta)drd\theta \\
&= \int_0^1\int_0^1 f\left(\sqrt{-2\log u}\cos(2\pi v), \sqrt{-2\log u}\sin(2\pi v)\right)dudv \\
&= \mathbb{E}f\left(\sqrt{-2\log U}\cos(2\pi V), \sqrt{-2\log U}\sin(2\pi V)\right).
\end{aligned}$$

For the simulation of a Gaussian random variable with mean μ and variance σ^2, it suffices to define $X = \mu + \sigma Y$, where $Y \simeq N(0, 1)$.

Simulation of a Poisson random variable A Poisson random variable with parameter λ is an $\mathbb{N}$-valued random variable such that

$$\mathbb{P}(X = n) = e^{-\lambda}\frac{\lambda^n}{n!}, \quad \text{for} \quad n \geq 0.$$

We shall see in Chapter 6 that whenever $\{T_i; i \geq 1\}$ is a sequence of i.i.d. random variables, all being exponential with parameter λ, then the law of $N_t = \sum_{n\geq 1} n\mathbf{1}_{\{T_1+\ldots+T_n\leq t<T_1+\ldots+T_{n+1}\}}$ is Poisson with parameter λt. Hence N_1 has the law which we want to simulate. On the other hand, any exponential random variable T_i can be written in the form $-\log(U_i)/\lambda$, where the $(U_i)_{i\geq 1}$ are mutually independent $\mathcal{U}(0, 1)$ random variables. Hence N_1 can be written

$$N_1 = \sum_{n\geq 1} n\mathbf{1}_{\{U_1U_2\cdots U_{n+1}<e^{-\lambda}\leq U_1U_2\cdots U_n\}}.$$

This gives an algorithm for the simulation of Poisson random variables.

The rejection method Suppose we wish to simulate a random variable with density f (e.g. with respect to Lebesgue measure on $\mathbb{R}^d$), and suppose that there is an easily simulable density g, such that, for all $x \in \mathbb{R}^d$,

$$f(x) \leq k\, g(x), \quad g(x) > 0 \Leftrightarrow f(x) > 0,$$

where k is a real constant. Define

$$\alpha(x) = \frac{f(x)}{k\, g(x)}$$

on the set $\{g(x) > 0\}$.

Proposition 3.2 *Let $(X_n, U_n)_{n\geq 1}$ be a sequence of independent random vectors where, for each $n \geq 1$, X_n and U_n are independent, X_n has the density g and $U_n \simeq \mathcal{U}(0,1)$. Let $N = \inf\{k \geq 1;\ U_k \leq \alpha(X_k)\}$ and $X = X_N$. The random variable X has the density f.*

Remark 3.3 *1. The probability of acceptance at the first step is*

$$\begin{aligned} p_1 &= \mathbb{P}(U_1 \leq \alpha(X_1)) \\ &= \int \mathbb{P}(U_1 \leq \alpha(x))\mathbb{P}_{X_1}(dx) \\ &= \int \alpha(x)g(x)dx \\ &= \frac{1}{k}, \end{aligned}$$

since U_1 and X_1 are independent.

If we wish to reduce the number of rejections while simulating X, we need to maximize the acceptance probability p_1, hence to minimize k. Given that f and g are probability densities and that $f \leq kg$, necessarily $k \geq 1$. Note that the number of rejections is limited if $f(x)/kg(x)$ is close to 1, that is, if the function g is similar to f.

2. *The above algorithm is still valid if X has a density f with respect to an arbitrary positive measure μ, which is bounded from above by kg, where g is the density with respect to μ of an easily simulable random variable Y. In other words,*

$$\mathbb{P}(X \in A) = \int_A f(x)\mu(dx) \leq \int_A kg(x)\mu(dx) = k\mathbb{P}(Y \in A).$$

If the law of X is supported by a discrete set E, we can choose for μ the counting measure of the points of E. The rejection method can be used for laws on a discrete set. In this case, $f(x) = \mathbb{P}(X = x)$.

Proof of Proposition 3.2 Note that the inequality $U_k \leq \alpha(X_k)$ will be satisfied after a finite number of steps. Indeed,

$$\begin{aligned} \mathbb{P}(\forall k \geq 1,\ X \neq X_k) &= \lim_{n\to\infty} \mathbb{P}(\cap_{k\leq n}\{X \neq X_k\}) \\ &= \lim_{n\to\infty} \mathbb{P}(\cap_{k\leq n}\{U_k > \alpha(X_k)\}) \\ &= \lim_{n\to\infty} \mathbb{P}(U_1 > \alpha(X_1))^n \\ &= \lim_{n\to\infty} (1 - p_1)^n = 0, \end{aligned}$$

since the random variables (X_k, U_k) are i.i.d. Consequently,

$$\begin{aligned}
\mathbb{P}[X \in A] &= \sum_{n\geq 1} \mathbb{P}[N = n, X \in A] \\
&= \sum_{n\geq 1} \mathbb{P}[\cap_{k\leq n-1}\{U_k > \alpha(X_k)\} \cap \{U_n \leq \alpha(X_n)\} \cap \{X_n \in A\}] \\
&= \sum_{n\geq 1} (1-p_1)^{n-1} \mathbb{P}[\{U_1 \leq \alpha(X_1)\} \cap \{X_1 \in A\}] \\
&= \frac{1}{p_1} \mathbb{P}[\{U_1 \leq \alpha(X_1)\} \cap \{X_1 \in A\}] \\
&= \mathbb{P}[X_1 \in A | U_1 \leq \alpha(X_1)].
\end{aligned}$$

The law of X is then the law of X_1, conditioned upon the acceptation set $\{U_1 \leq \alpha(X_1)\}$. From the independence of X_1 and U_1,

$$\begin{aligned}
\mathbb{P}[X \in A] &= \frac{1}{p_1} \int_A \mathbb{P}(U_1 \leq \alpha(x)) P_{X_1}(dx) \\
&= k \int_A \alpha(x) g(x) dx \\
&= \int_A f(x) dx.
\end{aligned}$$

□

For the simulation of other laws, or other simulation methods of the above laws, one can consult, among others, [7], [8], [13] and [35].

1.4 Variance reduction techniques

We have seen that the rate of convergence of the Monte Carlo method is of order $\sigma/\sqrt{n}$. Clearly, the convergence is accelerated if the variance is reduced. We now present several variance reduction methods.

Importance sampling Suppose that we try to compute $\mathbb{E}(g(X))$, where the law of X is $f(x)dx$ (on $\mathbb{R}$, for the sake of argument). We have

$$\mathbb{E}(g(X)) = \int_{\mathbb{R}} g(x) f(x) dx.$$

But if $\tilde{f}$ is the density of a probability such that $\tilde{f} > 0$, then one can rewrite $\mathbb{E}(g(X))$ as

$$\mathbb{E}(g(X)) = \int_{\mathbb{R}} \frac{g(x) f(x)}{\tilde{f}(x)} \tilde{f}(x) dx.$$

This means that $\mathbb{E}(g(X)) = \mathbb{E}\left(g(Y)f(Y)/\tilde{f}(Y)\right)$, where Y has the law $\tilde{f}(x)dx$. Hence, there is another method for computing $\mathbb{E}(g(X))$, using n simulations $Y_1, \ldots, Y_n$ of Y, and approximating $\mathbb{E}(g(X))$ by

$$\frac{1}{n}\left(\frac{g(Y_1)f(Y_1)}{\tilde{f}(Y_1)} + \ldots + \frac{g(Y_n)f(Y_n)}{\tilde{f}(Y_n)}\right).$$

If we let $Z = g(Y)f(Y)/\tilde{f}(Y)$, then this alternative method improves the convergence provided $\mathrm{var}(Z) < \mathrm{var}(g(X))$. It is easy to compute the variance of Z:

$$\mathrm{var}(Z) = \mathbb{E}(Z^2) - \mathbb{E}(Z)^2 = \int_{\mathbb{R}} \frac{g^2(x)f^2(x)}{\tilde{f}(x)}dx - \mathbb{E}(g(X))^2.$$

If $g(x) \geq 0$, it is easy to see that choosing $\tilde{f}(x) = g(x)f(x)/\mathbb{E}g(X)$ makes $\mathrm{var}(Z) = 0$. Of course, this relies on the fact that we can compute $\mathbb{E}(g(X))$ exactly.

This justifies the following heuristic: choose $\tilde{f}(x)$ as close as possible to $|g(x)f(x)|$, then normalize (divide by $\int \tilde{f}(x)dx$) so as to obtain a density of an easily simulable probability law. Of course, these constraints are largely contradictory.

Let us give one simple example. Suppose that we seek to compute

$$\int_0^1 \cos(\pi x/2)\, dx.$$

Let us replace the function cos by a polynomial of degree 2. Since the integrand is even and equals 0 at $x = 1$ and 1 at $x = 0$, it is natural to choose $\tilde{f}(x)$ of the form $\lambda(1 - x^2)$. If we normalize, we get $\tilde{f}(x) = 3(1 - x^2)/2$. If we compute the variances, we can verify that the method has reduced the variance by a factor of 100.

Control variate This method involves writing $\mathbb{E}(f(X))$ in the form

$$\mathbb{E}(f(X)) = \mathbb{E}(f(X) - h(X)) + \mathbb{E}(h(X)),$$

where $\mathbb{E}(h(X))$ can be explicitly computed, and $\mathrm{var}(f(X) - h(X))$ is significantly smaller than $\mathrm{var}(f(X))$. We then use a Monte Carlo method for the computation of $\mathbb{E}(f(X) - h(X))$ and a direct computation for $\mathbb{E}(h(X))$.

Let us start with a simple example. Suppose we wish to compute $\int_0^1 e^x dx$. Since near $x = 0$, $e^x \approx 1 + x$, we can write

$$\int_0^1 e^x dx = \int_0^1 (e^x - 1 - x)dx + \frac{3}{2}.$$

It is easy to see that the variance is significantly reduced.

In applications to finance (see Chapter 9), one needs to evaluate quantities of the type

$$C = \mathbb{E}\left(\left(e^{\sigma Z} - K\right)_+\right), \tag{1.1}$$

where Z is standard normal random variable and $x_+ = \max(0, x)$. Such a quantity is the price of a call option. Of course, in this precise case, there is an explicit formula for the above quantity, namely the celebrated Black–Scholes formula,

$$\mathbb{E}\left(\left(e^{\sigma Z} - K\right)_+\right) = e^{\sigma^2/2} F\left(\sigma - \frac{\log(K)}{\sigma}\right) - K F\left(-\frac{\log(K)}{\sigma}\right), \tag{1.2}$$

where

$$F(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-u^2/2} du.$$

However there are variants of this problem which can be solved only by the Monte Carlo method (see Chapter 9). Suppose that we wish to compute the above quantity by the Monte Carlo method, that is, we approximate that quantity by

$$C \simeq n^{-1}\left[\left(e^{\sigma Z_1} - K\right)_+ + \ldots + \left(e^{\sigma Z_n} - K\right)_+\right].$$

Suppose now that we wish to evaluate the price of a put option,

$$P = \mathbb{E}\left(\left(K - e^{\sigma Z}\right)_+\right), \tag{1.3}$$

hence

$$P \simeq n^{-1}\left[\left(K - e^{\sigma Z_1}\right)_+ + \ldots + \left(K - e^{\sigma Z_n}\right)_+\right].$$

At least whenever $K^2 << \exp(\sigma^2/2)$,

$$\mathrm{var}\left[\left(K - e^{\sigma Z}\right)_+\right] < \mathrm{var}\left[\left(e^{\sigma Z} - K\right)_+\right].$$

The put–call parity relationship (which follows from C and P, and the relation $x = x^+ - x^-$) says that

$$C - P = e^{-\sigma^2/2} - K,$$

hence we should instead compute P by a Monte Carlo procedure, and use the put–call parity relationship in order to get C, rather than computing C directly by Monte Carlo (see Exercise 5.9 below).

Antithetic variables Suppose we wish to compute

$$I = \int_0^1 f(x) dx.$$

Since $x \to 1 - x$ leaves the measure dx invariant on $[0, 1]$,

$$I = \frac{1}{2} \int_0^1 (f(x) + f(1 - x)) dx.$$

We can then compute I as follows. We simulate n i.i.d. $\mathcal{U}(0,1)$ random variables $U_1, \ldots, U_n$, and we approximate I by

$$I_{2n} = \frac{1}{n}\left(\frac{1}{2}(f(U_1) + f(1-U_1)) + \ldots + \frac{1}{2}(f(U_n) + f(1-U_n))\right)$$
$$= \frac{1}{2n}\left(f(U_1) + f(1-U_1) + \ldots + f(U_n) + f(1-U_n)\right).$$

If we compare this method with a direct Monte Carlo method after n simulations, we note that the approximation is improved provided

$$\mathbb{E}f(U)f(1-U) < \mathbb{E}f^2(U),$$

which holds true provided the random variables $f(U)$ and $f(1-U)$ are linearly independent.

The method can be generalized to higher dimensions, and to other transformations which leave the law of the random variable to be simulated invariant.

For example, if we try to compute the price of a put option (1.3), we can use the fact that the law of Z is identical to that of $-Z$ and reduce the variance by a factor of at least 2. Indeed, if $f(x) = [K - e^{\sigma x}]_+$, $\sigma > 0$, f is monotone decreasing, hence

$$\operatorname{var}\left(\frac{f(Z) + f(-Z)}{2}\right) = \frac{1}{2}\operatorname{var}(f(Z)) + \frac{1}{2}\operatorname{cov}(f(Z), f(-Z))$$
$$\leq \frac{1}{2}\operatorname{var}(f(Z)),$$

since

$$\operatorname{cov}(f(Z), f(-Z)) \leq \mathbb{E}\left([f(Z) - f(0)][f(-Z) - f(0)]\right)$$
$$\leq 0.$$

Stratification method This method is well known in the context of survey sample design. Suppose we seek to compute

$$I = \mathbb{E}(g(X)) = \int g(x)f(x)dx,$$

where X has the law $f(x)dx$. We start by decomposing I into

$$I = \sum_{i=1}^{m} I_i = \sum_{i=1}^{m} \mathbb{E}(\mathbf{1}_{\{X \in D_i\}} g(X)),$$

where D_i is a partition of the integration set. We then use n_i simulations for the computation of I_i. Define $\sigma_i^2 = \operatorname{var}(\mathbf{1}_{\{X \in D_i\}} g(X))$. Then the variance of the approximation is

$$\sum_{i=1}^{m} \frac{\sigma_i^2}{n_i}.$$

If we minimize this quantity with the constraint that $\sum_{i=1}^m n_i = n$ is fixed, we get $n_i = n\sigma_i / \sum_{i=1}^m \sigma_i$. The minimum equals $n^{-1} \left(\sum_{i=1}^m \sigma_i \right)^2$. We can show that it is smaller that the variance obtained with n simulations of a standard Monte Carlo procedure. Of course, one can rarely compute the σ_i, which limits the use of this technique (but we can estimates the σ_i via a Monte Carlo procedure!). To learn more about this procedure, see [10].

Mean value Suppose we wish to compute

$$\mathbb{E}(g(X, Y)) = \int g(x, y) f(x, y) dx dy,$$

where $f(x, y)dxdy$ is the law of the pair (X, Y).

If we let

$$h(x) = \frac{1}{m(x)} \int g(x, y) f(x, y) dy,$$

with $m(x) = \int f(x, y)dy$, it is easy to check that

$$\mathbb{E}(g(X, Y)) = \mathbb{E}(h(X)).$$

Indeed, the law of X is $m(x)dx$, hence

$$\mathbb{E}(h(X)) = \int m(x)h(x)dx = \int dx \int g(x, y) f(x, y) dy = \mathbb{E}(g(X, Y)).$$

On the other hand, interpreting $h(X)$ as a conditional expectation, we can show that

$$\text{var}(h(X)) \leq \text{var}(g(X, Y)).$$

Consequently, if we can compute the function h explicitly, it is preferable to use a Monte Carlo procedure for $h(X)$.

Remark 4.1 *We wrote in the introduction to this chapter that the Monte Carlo method is particularly well suited to the computation of multiple integrals. We shall see a typical example of such a situation, for a mathematical finance problem, in Exercise 7.5. of Chapter 9.*

1.5 Exercises

Exercise 5.1 *Let X be a geometric random variable with parameter p, that is, $\mathbb{P}(X = k) = p(1 - p)^{k-1}, k \geq 1$.*

1. *Describe a method for simulating X based on a sequence of Bernoulli trials.*
2. *Give another method for simulating this law based on the formula $\mathbb{P}(X > k) = (1 - p)^k$, $k \geq 0$, and compare the two methods.*

Exercise 5.2 *1. Describe a standard method for simulating the Gaussian* $N(0, 1)$ *law.*

2. Propose a rejection algorithm for the simulation of a Gaussian random variable, based upon the simulation of doubly exponential random variables with density $(\lambda/2)\exp(-\lambda|x|)$.

3. Let X *and* Y *be two independent random variables, both exponential with parameter 1.*

(a) Give the conditional law of X, *given that* $\{Y > (1-X)^2/2\}$.

(b) Let Z *be a random variable having the above conditional law, and* S *an independent random variable taking the values* ± 1 *with probability* $1/2$. *Give the law of* SZ.

(c) Deduce another method for simulating the Gaussian $N(0, 1)$ *law.*

Exercise 5.3 *A process* $\{X(t);\ t \geq 0\}$ *with continuous trajectories is said to be a Brownian motion if it possesses the two following properties:*

(i) For any $n \geq 1$, $0 = t_0 < t_1 < t_2 < \ldots < t_n$, *the random variables* $X(t_k) - X(t_{k-1})(1 \leq k \leq n)$ *are mutually independent (we say that X(t) has independent increments).*

(ii) $X(0) = 0$ *and the law of* $X(t+h) - X(t)$ *is the Gaussian law* $N(0, h)$, *for all* $t \geq 0$, $h > 0$.

1. Propose a method for simulating $\{X(kh);\ k \geq 1\}$, *for a given* $h > 0$.

2. Give the conditional law of $X(t)$, *given that* $X(t-a) = x$ *and* $X(t+a) = y$. *Deduce a method for simulating* $\{X(kh/2);\ k \geq 1\}$ *which avoids the need to redo the simulations of part 1.*

Exercise 5.4 *Let* (X_1, X_2) *be a Gaussian random vector, with correlation coefficient* ρ *and such that, for* $i = 1, 2$, *the random variable* X_i *has the law* $N(\mu_i, \sigma_i^2)$.

1. Show that if (Y_1, Y_2) *is a pair of* $N(0, 1)$ *independent random variables, then the pair* $Z_1 = \mu_1 + \sigma_1 Y_1$, $Z_2 = \mu_2 + \sigma_2(\rho Y_1 + \sqrt{1-\rho^2}Y_2)$ *has the same law as* (X_1, X_2). *Deduce a method for simulating this random vector.*

2. Generalize to the case of an arbitrary dimension.

Exercise 5.5 *Let* X *denote a random variable with the distribution function* F. *Assume that* F *is one-to-one, and denote its inverse by* F^{-1}.

1. Give a method for simulating X *conditionally upon* $X > m$, *based on a rejection method. Discuss the efficiency of the method. What happens when* m *is large?*

2. *For a $\mathcal{U}(0,1)$ random variable U, define*

$$Z = F^{-1}\left(F(m) + (1 - F(m))U\right).$$

Compute the distribution function of Z and deduce a method of simulating X, conditionally upon $X > m$. Compare with the above rejection method.

3. *Generalize the previous method to the case where one seeks to simulate X conditionally upon $a < X < b$.*

4. *Suppose we now try to simulate a Gaussian $N(\mu, \sigma^2)$ random variable X, conditionally upon $X > m$. Show that we can restrict ourselves to the case of a standard normal random variable, provided we modify the value of m.*

5. *Propose, for the problem of part 4, a rejection method based upon a translated exponential law with the density $\theta e^{-\theta(x-m)}\mathbf{1}_{\{x > m\}}$. How should one choose the parameter θ?*

Exercise 5.6 (Importance sampling) *Suppose we wish to compute by a Monte Carlo method the quantity*

$$p_\ell = \mathbb{P}(X \in [\ell,\ \ell + 1]),$$

where X is an exponential random variable with parameter 1.

1. *Give the standard estimator of p_ℓ and compute its variance.*

2. *Propose an importance sampling method, such that the new simulations all belong to the interval $[\ell,\ \ell + 1]$. Compute the variance of this new estimator and discuss the case of large values of ℓ.*

Exercise 5.7 (Variance reduction)

1. *Propose an importance sampling method for the computation of*

$$I = \mathbb{E}\left(\mathbf{1}_{\{X > 0\}} \exp \beta X\right),$$

where X is a Gaussian $N(0,1)$ random variable and $\beta = 5$.

2. *Propose a control variate method for the same computation.*

3. *Improve the method with the help of an antithetic variable method.*

Exercise 5.8 *The aim of this exercise is to prove that the method of antithetic variables reduces the variance whenever we have a function which is monotone in each of its variables.*

1. *Suppose that f and g are both bounded and increasing from $\mathbb{R}$ into $\mathbb{R}$. Show that for any real-valued random variables X and Y,*

$$\mathbb{E}(f(X)g(X)) + \mathbb{E}(f(Y)g(Y)) \geq \mathbb{E}(f(X)g(Y)) + \mathbb{E}(f(Y)g(X)).$$

Deduce that for any real random variable X,

$$\mathbb{E}(f(X)g(X)) \geq \mathbb{E}(f(X))\,\mathbb{E}(g(X)) \Leftrightarrow \operatorname{cov}(f(X), g(X)) \geq 0.$$

2. *Show that if $X_1, \ldots, X_n$ are mutually independent real random variables,*

$$\mathbb{E}(f(X_1, \ldots, X_n)g(X_1, \ldots, X_n)|X_n) = \Phi(X_n),$$

where Φ is a function to be expressed as an expectation. Deduce that whenever f and g are increasing in each of their arguments,

$$\mathbb{E}(f(X_1, \ldots, X_n)g(X_1, \ldots, X_n)) \geq \mathbb{E}(f(X_1, \ldots, X_n))\,\mathbb{E}(g(X_1, \ldots, X_n)).$$

3. *Let h be a mapping from $[0,1]^n$ into $\mathbb{R}$, which is monotone in each of its arguments, and let $U_1, \ldots, U_n$ be independent $\mathcal{U}(0,1)$ random variables. Show that*

$$\operatorname{cov}(h(U_1, \ldots, U_n), h(1 - U_1, \ldots, 1 - U_n)) \leq 0,$$

and show that the method of antithetic random variables reduces the variance in this case.

Exercise 5.9 (Programming) *Recall the formula (1.1) for the price of a call option, and (1.3) for the price of a put option. Deduce from the identity $x = x^+ - (-x)^+$ the put–call parity relationship*

$$C - P = \mathbb{E}e^{\sigma Z} - K,$$

where the expectation $\mathbb{E}e^{\sigma Z}$ can be computed explicitly and equals $\exp(\sigma^2/2)$. Deduce from this identity a control variate method, and show that it reduces the variance.

Since Z and $-Z$ have the same law, one can apply a method of antithetic random variables to the two Monte Carlo computations of the call and of the put.

Choose for the simulation $\sigma = 1.5$ and $K = 1$. Do the Monte Carlo computations with sample sizes $N = 1000, 10\,000$ and $100\,000$. For each computation, give the estimate deduced from the Monte Carlo simulations, and a 95 % confidence interval, based on the central limit theorem and an estimate of the variance.

1. *Compute the value deduced from the Black–Scholes formula (1.2).*

2. *Compute C by a Monte Carlo procedure, using first the formula (1.1), and then the put–call parity relationship and (1.3) for the computation of P by a Monte Carlo procedure.*

3. *Repeat the same two computations, using an antithetic variable method.*