Chapter 13

Monte Carlo Simulation Methods

Monte Carlo methods consist of solving mathematical problems using random numbers. The term '*Monte Carlo*' was apparently coined by physicists Ulam and von Neumann at Los Alamos in 1940 and refers to gambling casinos in Monaco[1]. Until the end of the eighties, Monte Carlo methods were principally used to calculate numerical integration[2] including mathematical expectations. More recently, the Monte Carlo method designates all numerical methods that involves stochastic simulation and consider random experiments on a computer.

This chapter is divided into three sections. In the first section, we present the different approaches to generate random numbers. Section two extends simulation methods when we manipulate stochastic processes. Finally, the last section is dedicated to Monte Carlo and quasi-Monte Carlo methods.

13.1 Random variate generation

Any Monte Carlo method is based on series of random variates that are independent and identically distributed (*iid*) according to a given probability distribution $\mathbf{F}$. As we will see later, it can be done by generating uniform random numbers. This is why numerical programming softwares already contain uniform random number generators. However, true randomness is impossible to simulate with a computer. In practice, only sequences of '*pseudorandom*' numbers can be produced with statistical properties that are close from those obtained with *iid* random variables.

13.1.1 Generating uniform random numbers

A first idea is to build a pseudorandom sequence $\mathcal{S}$ and repeat this sequence as often as necessary. For instance, for simulating 10 uniform random numbers, we can set $\mathcal{S} = \{0, 0.5, 1\}$ and repeat this sequence four times. In this case, the 10 random numbers are:

$$\{0, 0.5, 1, 0, 0.5, 1, 0, 0.5, 1, 0\}$$

We notice that the period length of this sequence is three. The quality of the pseudorandom number generator depends on the period length, which should be large in order to avoid duplication and serial correlation. If we calculate the second moment of $\mathcal{S}$, we do not obtain the variance of a uniform random variable $\mathcal{U}_{[0,1]}$. A good pseudorandom number generator should therefore pass standard adequacy tests.

[1] Monte Carlo is one of the four quarters of Monaco and houses the famous casino.
[2] In this case, we speak about Monte Carlo integration methods.

The most famous and used algorithm is the linear congruential generator (LCG):

$$\begin{aligned} x_n &= (a \cdot x_{n-1} + c) \bmod m \\ u_n &= x_n / m \end{aligned}$$

where a is the multiplicative constant, c is the additive constant and m is the modulus (or the order of the congruence). To initialize the algorithm, we have to define the initial number x_0, called the seed[3]. $\{x_1, x_2, \ldots, x_n\}$ is a sequence of pseudorandom integer numbers $(0 \leq x_n < m)$ whereas $\{u_1, u_2, \ldots, u_n\}$ is a sequence of uniform random variates. We can show that the maximum period[4] is m and can be only achieved for some specific values of a, c and m. The quality of the random number generator will then depend on the values of the parameters.

Example 133 *If we consider that* $a = 3$, $c = 0$, $m = 11$ *and* $x_0 = 1$, *we obtain the following sequence:*

$$\{1, 3, 9, 5, 4, 1, 3, 9, 5, 4, 1, 3, 9, 5, 4, \ldots\}$$

The period length is only five, meaning that only five uniform random variates can be generated: 0.09091, 0.27273, 0.81818, 0.45455 *and* 0.36364.

The minimal standard LCG proposed by Lewis *et al.* (1969) is defined by $a = 7^5$, $c = 0$ and $m = 2^{31} - 1$. In Table 13.1, we report two sequences generated with the seed values 1 and 123 456. This generator is widely used in numerical programming languages. However, its period length is equal to $m - 1 = 2^{31} - 2 \approx 2.15 \times 10^9$, which can be judged as insufficient for some modern Monte Carlo applications. For instance, if we consider the LDA model in operational risk with a Poisson distribution $\mathcal{P}(1\,000)$, we need approximately 10^{10} random numbers for drawing the severity loss if the number of Monte Carlo simulations is set to ten million. Another drawback is that LCG methods may exhibit lattice structures. For instance, Figure 13.1 shows the dependogram between u_{n-1} and u_n when $a = 10$, $c = 0$ and $m = 2^{31} - 1$.

TABLE 13.1: Simulation of 10 uniform pseudorandom numbers

n	x_n	u_n	x_n	u_n
0	1	0.000000	123 456	0.000057
1	16 807	0.000008	2 074 924 992	0.966212
2	282 475 249	0.131538	277 396 911	0.129173
3	1 622 650 073	0.755605	22 885 540	0.010657
4	984 943 658	0.458650	237 697 967	0.110687
5	1 144 108 930	0.532767	670 147 949	0.312062
6	470 211 272	0.218959	1 772 333 975	0.825307
7	101 027 544	0.047045	2 018 933 935	0.940139
8	1 457 850 878	0.678865	1 981 022 945	0.922486
9	1 458 777 923	0.679296	466 173 527	0.217079
10	2 007 237 709	0.934693	958 124 033	0.446161

Nowadays, with a 64-bit computer, the maximum period of a LCG algorithm is $2^{64} - 1 \approx 1.85 \times 10^{19}$. To obtain a larger period length, one can use more sophisticated methods. For

[3] If the seed is not specified, programming softwares generally use the clock of the computer to generate the initial value.

[4] It is equal to $m - 1$ if $c = 0$.

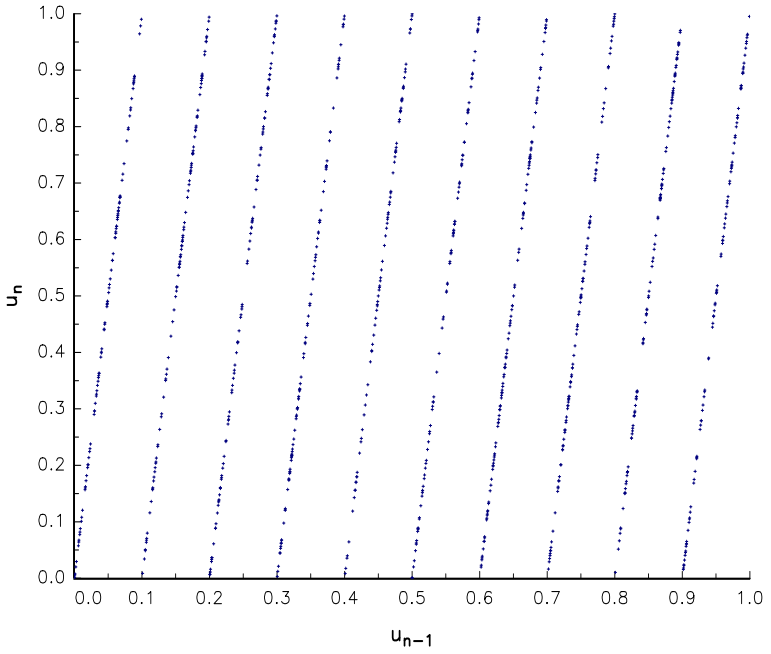

FIGURE 13.1: Lattice structure of the linear congruential generator

instance, multiple recursive generators are based on the following transition equation:

$$x_n = \left(\sum_{i=1}^{k} a_i \cdot x_{n-i} + c \right) \bmod m$$

To obtain a bigger period, we can also combine LCG algorithms with different periods. For instance, the famous MRG32k3a generator of L'Ecuyer (1999) uses two 32-bit multiple recursive generators:

$$\begin{cases} x_n = (1403580 \cdot x_{n-2} - 810728 \cdot x_{n-3}) \bmod m_1 \\ y_n = (527612 \cdot y_{n-1} - 1370589 \cdot y_{n-3}) \bmod m_2 \end{cases}$$

where $m_1 = 2^{32} - 209$ and $m_2 = 2^{32} - 22853$. The uniform random variate is then equal to:

$$u_n = \frac{x_n - y_n + \mathbb{1}\{x_n \leq y_n\} \cdot m_1}{m_1 + 1}$$

L'Ecuyer (1999) showed that the period length of this generator is equal to $2^{191} \approx 3 \times 10^{57}$.

13.1.2 Generating non-uniform random numbers

We now consider X a random variable whose distribution function is noted $\mathbf{F}$. There are many ways to simulate X, but all of them are based on uniform random variates.

13.1.2.1 Method of inversion

Continuous random variables We assume that $\mathbf{F}$ is continuous. Let $Y = \mathbf{F}(X)$ be the integral transform of X. Its cumulative distribution function $\mathbf{G}$ is equal to:

$$\begin{aligned}\mathbf{G}(y) &= \Pr\{Y \leq y\} \\ &= \Pr\{\mathbf{F}(X) \leq y\} \\ &= \Pr\left\{X \leq \mathbf{F}^{-1}(y)\right\} \\ &= \mathbf{F}\left(\mathbf{F}^{-1}(y)\right) \\ &= y\end{aligned}$$

where $\mathbf{G}(0) = 0$ and $\mathbf{G}(1) = 1$. We deduce that $\mathbf{F}(X)$ has a uniform distribution $\mathcal{U}_{[0,1]}$. It follows that if U is a uniform random variable, then $\mathbf{F}^{-1}(U)$ is a random variable whose distribution function is $\mathbf{F}$. To simulate a sequence of random variates $\{x_1, \ldots, x_n\}$, we can simulate a sequence of uniform random variates $\{u_1, \ldots, u_n\}$ and apply the transform $x_i \leftarrow \mathbf{F}^{-1}(u_i)$.

Example 134 *If we consider the generalized uniform distribution $\mathcal{U}_{[a,b]}$, we have $\mathbf{F}(x) = (x-a)/(b-a)$ and $\mathbf{F}^{-1}(u) = a + (b-a)u$. The simulation of random variates x_i is deduced from the uniform random variates u_i by using the following transform:*

$$x_i \leftarrow a + (b-a)\,u_i$$

Example 135 *In the case of the exponential distribution $\mathcal{E}(\lambda)$, we have $\mathbf{F}(x) = 1 - \exp(-\lambda x)$. We deduce that:*

$$x_i \leftarrow -\frac{\ln(1-u_i)}{\lambda}$$

Since $1-U$ is also a uniform distributed random variable, we have:

$$x_i \leftarrow -\frac{\ln(u_i)}{\lambda}$$

Example 136 *In the case of the Pareto distribution $\mathcal{P}(\alpha, x_-)$, we have $\mathbf{F}(x) = 1 - (x/x_-)^{-\alpha}$ and $\mathbf{F}^{-1}(u) = x_-(1-u)^{-1/\alpha}$. We deduce that:*

$$x_i \leftarrow \frac{x_-}{(1-u_i)^{1/\alpha}}$$

The method of inversion is easy to implement when we know the analytical expression of $\mathbf{F}^{-1}$. When it is not the case, we use the Newton-Raphson algorithm:

$$x_i^{m+1} = x_i^m + \frac{u_i - \mathbf{F}(x_i^m)}{f(x_i^m)}$$

where x_i^m is the solution of the equation $\mathbf{F}(x) = u$ at the iteration m. For instance, if we apply this algorithm to the Gaussian distribution $\mathcal{N}(0,1)$, we have:

$$x_i^{m+1} = x_i^m + \frac{u_i - \Phi(x_i^m)}{\phi(x_i^m)}$$

Discrete random variables In the case of a discrete probability distribution $\{(x_1, p_1), (x_2, p_2), \ldots, (x_n, p_n)\}$ where $x_1 < x_2 < \ldots < x_n$, we have:

$$\mathbf{F}^{-1}(u) = \begin{cases} x_1 & \text{if} \quad 0 \leq u \leq p_1 \\ x_2 & \text{if} \quad p_1 < u \leq p_1 + p_2 \\ \vdots & \\ x_n & \text{if} \quad \sum_{k=1}^{n-1} p_k < u \leq 1 \end{cases}$$

In Figure 13.2, we illustrate the method of inversion when the random variable is discrete. We assume that:

x_i	1	2	4	6	7	9	10
p_i	10%	20%	10%	5%	20%	30%	5%
$\mathbf{F}(x_i)$	10%	30%	40%	45%	65%	95%	100%

Because the cumulative distribution function is not continuous, the inverse function is a step function. If we suppose that the uniform random number is 0.5517, we deduce that the corresponding random number for the variable X is equal to 7.

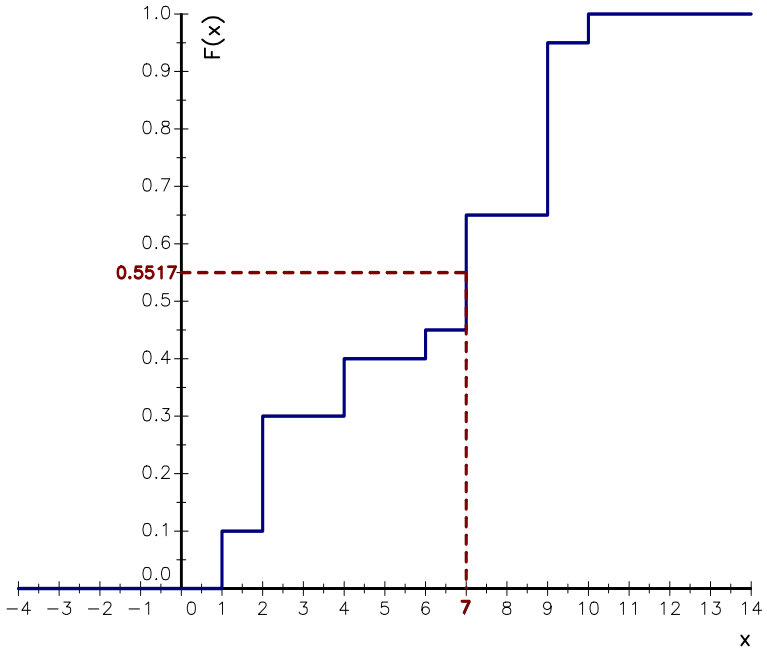

FIGURE 13.2: Inversion method when X is a discrete random variable

Example 137 *If we apply the method of inversion to the Bernoulli distribution* $\mathcal{B}(p)$*, we have:*

$$x \leftarrow \begin{cases} 0 & \textit{if} \quad 0 \leq u \leq 1-p \\ 1 & \textit{if} \quad 1-p < u \leq 1 \end{cases}$$

or:

$$x \leftarrow \begin{cases} 1 & \textit{if} \quad u \leq p \\ 0 & \textit{if} \quad u > p \end{cases}$$

Piecewise distribution functions A piecewise distribution function is defined as follows:

$$\mathbf{F}(x) = \mathbf{F}_m(x) \qquad \text{if } x \in \left]x_{m-1}^\star, x_m^\star\right]$$

where $x_m^\star$ are the knots of the piecewise function and:

$$\mathbf{F}_{m+1}(x_m^\star) = \mathbf{F}_m(x_m^\star)$$

In this case, the simulated value x_i is obtained using a search algorithm:

$$x_i \leftarrow \mathbf{F}_m^{-1}(u_i) \qquad \text{if } \mathbf{F}\left(x_{m-1}^\star\right) < u_i \leq \mathbf{F}(x_m^\star)$$

This means that we have first to calculate the value of $\mathbf{F}(x)$ for all the knots in order to determine which inverse function $\mathbf{F}_m^{-1}$ will be apply.

Let us consider the piecewise exponential model described on page 202. We reiterate that the survival function has the following expression:

$$\mathbf{S}(t) = \mathbf{S}\left(t_{m-1}^\star\right) e^{-\lambda_m\left(t - t_{m-1}^\star\right)} \qquad \text{if } t \in \left]t_{m-1}^\star, t_m^\star\right]$$

We know that $\mathbf{S}(\boldsymbol{\tau}) \sim U$. It follows that:

$$t_i \leftarrow t_{m-1}^\star + \frac{1}{\lambda_m} \ln \frac{\mathbf{S}\left(t_{m-1}^\star\right)}{u_i} \qquad \text{if } \mathbf{S}\left(t_m^\star\right) < u_i \leq \mathbf{S}\left(t_{m-1}^\star\right)$$

Example 138 *We model the default time $\boldsymbol{\tau}$ with the piecewise exponential model and the following parameters:*

$$\lambda = \begin{cases} 5\% & \textit{if } t \textit{ is less or equal than one year} \\ 8\% & \textit{if } t \textit{ is between one and five years} \\ 12\% & \textit{if } t \textit{ is larger than five years} \end{cases}$$

We have $\mathbf{S}(0) = 1$, $\mathbf{S}(1) = 0.9512$ and $\mathbf{S}(5) = 0.6907$. We deduce that:

$$t_i \leftarrow \begin{cases} 0 + (1/0.05) \cdot \ln(1/u_i) & \text{if } u_i \in [0.9512, 1] \\ 1 + (1/0.08) \cdot \ln(0.9512/u_i) & \text{if } u_i \in [0.6907, 0.9512[\\ 5 + (1/0.12) \cdot \ln(0.6907/u_i) & \text{if } u_i \in [0, 0.6907[\end{cases}$$

In Table 13.2, we have reported five simulations t_i of the default time $\boldsymbol{\tau}$. For each simulation, we indicate the values taken by $t_{m-1}^\star$, $\mathbf{S}\left(t_{m-1}^\star\right)$ and λ_m.

TABLE 13.2: Simulation of the piecewise exponential model

u_i	$t_{m-1}^\star$	$S\left(t_{m-1}^\star\right)$	λ_m	t_i
0.9950	0	1.0000	0.05	0.1003
0.3035	5	0.6907	0.12	11.8531
0.5429	5	0.6907	0.12	7.0069
0.9140	1	0.9512	0.08	1.4991
0.7127	1	0.9512	0.08	4.6087

13.1.2.2 Method of transformation

Let $\{Y_1, Y_2, \ldots\}$ be a vector of independent random variables. The simulation of the random variable $X = g(Y_1, Y_2, \ldots)$ is straightforward if we know how to easily simulate the random variables Y_i. We notice that the inversion method is a particular case of the transform method, because we have:

$$X = g(U) = \mathbf{F}^{-1}(U)$$

Example 139 *The Binomial random variable is the sum of n iid Bernoulli random variables:*

$$\mathcal{B}(n, p) = \sum_{i=1}^{n} \mathcal{B}_i(p)$$

We can therefore simulate the Binomial random variate x using n uniform random numbers:

$$x = \sum_{i=1}^{n} \mathbb{1}\{u_i \leq p\}$$

If we would like to simulate the chi-squared random variable $\chi^2(\nu)$, we can use the following relationship:

$$\chi^2(\nu) = \sum_{i=1}^{\nu} \chi_i^2(1) = \sum_{i=1}^{\nu} (\mathcal{N}_i(0, 1))^2$$

We can therefore simulate the $\chi^2(\nu)$ random variate with ν independent Gaussian random numbers $\mathcal{N}(0, 1)$. For that, we generally use the Box-Muller algorithm, which states that if U_1 and U_2 are two independent uniform random variables, then X_1 and X_2 defined by:

$$\begin{cases} X_1 = \sqrt{-2 \ln U_1} \cdot \cos(2\pi U_2) \\ X_2 = \sqrt{-2 \ln U_1} \cdot \sin(2\pi U_2) \end{cases}$$

are independent and follow the Gaussian distribution $\mathcal{N}(0, 1)$.

Remark 149 *To simulate a Student's t random variate x with ν degrees of freedom, we need $\nu + 1$ normal independent random variables n_i:*

$$x \longleftarrow \frac{n_{\nu+1}}{\sqrt{\nu^{-1} \sum_{i=1}^{\nu} n_i^2}}$$

However, this method is not efficient and we generally prefer to use the Bailey algorithm based on the polar transformation[5]*.*

On page 339, we have seen that if N_t is a Poisson process with intensity λ, the duration T between two consecutive events is an exponential distributed random variable. We have:

$$\Pr(T \leq t) = 1 - e^{-\lambda t}$$

Since the durations are independent, we have:

$$T_1 + T_2 + \ldots + T_n = \sum_{i=1}^{n} E_i$$

[5] This method is presented on page 887.

where $E_i \sim \mathcal{E}(\lambda)$. Because the Poisson random variable is the number of events that occur in the unit interval of time, we also have:

$$\begin{aligned} X &= \max\{n : T_1 + T_2 + \ldots + T_n \leq 1\} \\ &= \max\left\{n : \sum_{i=1}^{n} E_i \leq 1\right\} \end{aligned}$$

We notice that:

$$\begin{aligned} \sum_{i=1}^{n} E_i &= -\frac{1}{\lambda}\sum_{i=1}^{n} \ln U_i \\ &= -\frac{1}{\lambda}\ln\prod_{i=1}^{n} U_i \end{aligned}$$

where U_i are *iid* uniform random variables. We deduce that:

$$\begin{aligned} X &= \max\left\{n : -\frac{1}{\lambda}\ln\prod_{i=1}^{n} U_i \leq 1\right\} \\ &= \max\left\{n : \prod_{i=1}^{n} U_i \geq e^{-\lambda}\right\} \end{aligned}$$

We can then simulate the Poisson random variable with the following algorithm:

1. set $n = 0$ and $p = 1$;
2. calculate $n = n + 1$ and $p = p \cdot u_i$ where u_i is a uniform random variate;
3. if $p \geq e^{-\lambda}$, go back to step 2; otherwise, return $X = n - 1$.

13.1.2.3 Rejection sampling

Following Devroye (1986), $\mathbf{F}(x)$ and $\mathbf{G}(x)$ are two distribution functions such that $f(x) \leq cg(x)$ for all x with $c > 1$. We note $X \sim \mathbf{G}$ and consider an independent uniform random variable $U \sim \mathcal{U}_{[0,1]}$. Then, the conditional distribution function of X given that $U \leq f(X)/(cg(X))$ is $\mathbf{F}(x)$.

Let us introduce the random variables B and Z:

$$\begin{aligned} B &= \mathbb{1}\left\{U \leq \frac{f(X)}{cg(X)}\right\} \\ Z &= X \left| U \leq \frac{f(X)}{cg(X)}\right. \end{aligned}$$

We have:

$$\begin{aligned} \Pr\{B = 1\} &= \Pr\left\{U \leq \frac{f(X)}{cg(X)}\right\} \\ &= \mathbb{E}\left[\frac{f(X)}{cg(X)}\right] \\ &= \int_{-\infty}^{+\infty} \frac{f(x)}{cg(x)} g(x)\,\mathrm{d}x \\ &= \frac{1}{c}\int_{-\infty}^{+\infty} f(x)\,\mathrm{d}x \\ &= \frac{1}{c} \end{aligned}$$

The distribution function of Z is defined by:

$$\Pr\{Z \leq x\} = \Pr\left\{X \leq x \middle| U \leq \frac{f(X)}{cg(X)}\right\}$$

We deduce that:

$$\begin{aligned}
\Pr\{Z \leq x\} &= \frac{\Pr\left\{X \leq x, U \leq \dfrac{f(X)}{cg(X)}\right\}}{\Pr\left\{U \leq \dfrac{f(X)}{cg(X)}\right\}} \\
&= c\int_{-\infty}^{x}\int_{0}^{f(x)/(cg(x))} g(x)\,\mathrm{d}u\,\mathrm{d}x \\
&= c\int_{-\infty}^{x}\frac{f(x)}{cg(x)}g(x)\,\mathrm{d}x \\
&= \int_{-\infty}^{x} f(x)\,\mathrm{d}x \\
&= \mathbf{F}(x)
\end{aligned}$$

This proves that $Z \sim \mathbf{F}$. From this theorem, we deduce the following acceptance-rejection algorithm:

1. generate two independent random variates x and u from $\mathbf{G}$ and $\mathcal{U}_{[0,1]}$;
2. calculate v as follows:
$$v = \frac{f(x)}{cg(x)}$$
3. if $u \leq v$, return x ('*accept*'); otherwise, go back to step 1 ('*reject*').

The underlying idea of this algorithm is then to simulate the distribution function $\mathbf{F}$ by assuming that it is easier to generate random numbers from $\mathbf{G}$, which is called the proposal distribution. However, some of these random numbers must be '*rejected*', because the function $c \cdot g(x)$ '*dominates*' the density function $f(x)$.

Remark 150 *We notice that the number of iterations N needed to successfully generate Z has a geometric distribution $\mathcal{G}(p)$, where $p = \Pr\{B = 1\} = c^{-1}$ is the acceptance ratio. We deduce that the average number of iterations is equal to $\mathbb{E}[N] = 1/p = c$. In order to maximize the efficiency (or the acceptance ratio) of the algorithm, we have to choose the constant c such that:*

$$c = \sup_x \frac{f(x)}{g(x)}$$

Let us consider the normal distribution $\mathcal{N}(0,1)$. We use the Cauchy distribution function as the proposal distribution, whose probability density function is given by:

$$g(x) = \frac{1}{\pi(1+x^2)}$$

We can show that:

$$\phi(x) \leq \frac{\sqrt{2\pi}}{e^{0.5}}g(x)$$

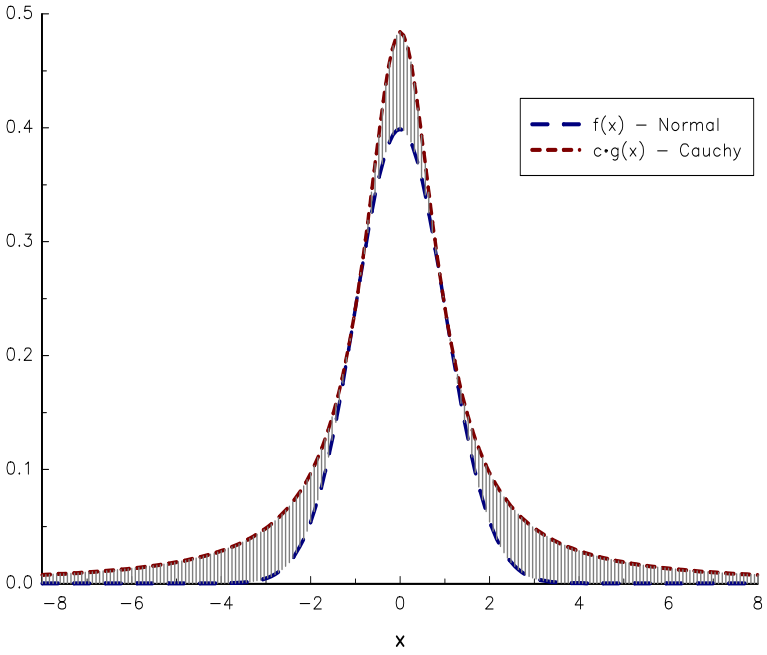

FIGURE 13.3: Rejection sampling applied to the normal distribution

meaning that $c \approx 1.52$. In Figure 13.3, we report the functions $f(x) = \phi(x)$ and $c \cdot g(x)$. The goal of the acceptance-rejection algorithm is to '*eliminate*' the random numbers, which are located in the cross-hatched region. Concerning the Cauchy distribution, we have:

$$\mathbf{G}(x) = \frac{1}{2} + \frac{1}{\pi} \arctan x$$

and:

$$\mathbf{G}^{-1}(u) = \tan\left(\pi\left(u - \frac{1}{2}\right)\right)$$

Therefore, we deduce that the following algorithm for simulating the distribution function $\mathcal{N}(0,1)$:

1. generate two independent uniform random variates u_1 and u_2 and set:

$$x \leftarrow \tan\left(\pi\left(u_1 - \frac{1}{2}\right)\right)$$

2. calculate v as follows:

$$\begin{aligned} v &= \frac{e^{0.5}\phi(x)}{\sqrt{2\pi}g(x)} \\ &= \frac{\left(1+x^2\right)}{2e^{(x^2-1)/2}} \end{aligned}$$

3. if $u_2 \leq v$, accept x; otherwise, go back to step 1.

To illustrate this algorithm, we have simulated six Gaussian distributed random variates in Table 13.3. We notice that four simulations have been rejected. Using 1 000 simulations of

Cauchy random variates, we obtained the density given in Figure 13.4, which is very close to the exact probability density function. In our case, we accept 683 simulations, meaning that the acceptance ratio[6] is 68.3%.

TABLE 13.3: Simulation of the standard Gaussian distribution using the acceptance-rejection algorithm

u_1	u_2	x	v	test	z
0.9662	0.1291	9.3820	0.0000	reject	
0.0106	0.1106	−30.0181	0.0000	reject	
0.3120	0.8253	−0.6705	0.9544	accept	−0.6705
0.9401	0.9224	5.2511	0.0000	reject	
0.2170	0.4461	−1.2323	0.9717	accept	−1.2323
0.6324	0.0676	0.4417	0.8936	accept	0.4417
0.6577	0.1344	0.5404	0.9204	accept	0.5404
0.1596	0.6670	−1.8244	0.6756	accept	−1.8244
0.4183	0.3872	−0.2625	0.8513	accept	−0.2625
0.9625	0.0752	8.4490	0.0000	reject	

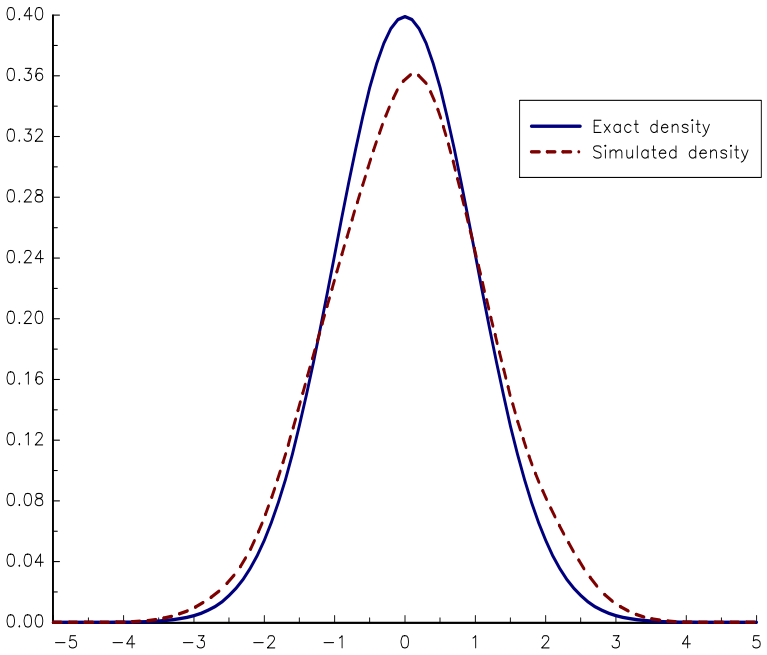

FIGURE 13.4: Comparison of the exact and simulated densities

Remark 151 *The discrete case is analogous to the continuous case. Let* $p(k)$ *and* $q(k)$ *be the probability mass function of* Z *and* X *such that* $p(k) \leq cq(k)$ *for all* k *with* $c \geq 1$. *We consider an independent uniform random variable* $U \sim \mathcal{U}_{[0,1]}$. *Then, the conditional pmf of* X *given that* $U \leq p(X) / (cq(X))$ *is the pmf* $p(k)$ *of* Z.

[6]The theoretical acceptance ratio is equal to $1/1.52 \approx 65.8\%$.

13.1.2.4 Method of mixtures

A finite mixture can be decomposed as a weighted sum of distribution functions. We have:

$$\mathbf{F}(x) = \sum_{k=1}^{n} \pi_k \cdot \mathbf{G}_k(x)$$

where $\pi_k \geq 0$ and $\sum_{k=1}^{n} \pi_k = 1$. We deduce that the probability density function is:

$$f(x) = \sum_{k=1}^{n} \pi_k \cdot g_k(x)$$

To simulate the probability distribution $\mathbf{F}$, we introduce the random variable B, whose probability mass function is defined by:

$$p(k) = \Pr\{B = k\} = \pi_k$$

It follows that:

$$\mathbf{F}(x) = \sum_{k=1}^{n} \Pr\{B = k\} \cdot \mathbf{G}_k(x)$$

We deduce the following algorithm:

1. generate the random variate b from the probability mass function $p(k)$;
2. generate the random variate x from the probability distribution $\mathbf{G}_b(x)$.

Example 140 *We assume that the default time $\boldsymbol{\tau}$ follows the hyper-exponential model:*

$$f(t) = \pi \cdot \lambda_1 e^{-\lambda_1 t} + (1-\pi) \cdot \lambda_2 e^{-\lambda_2 t}$$

To simulate this model, we consider the following algorithm:

1. we generate u and v two independent uniform random numbers;
2. we have:
$$b \leftarrow \begin{cases} 1 & \text{if } u \leq \pi \\ 2 & \text{otherwise} \end{cases}$$
3. the simulated value of $\boldsymbol{\tau}$ is:
$$t \leftarrow \begin{cases} -\lambda_1 \ln v & \text{if } b = 1 \\ -\lambda_2 \ln v & \text{if } b = 2 \end{cases}$$

Remark 152 *The previous approach can be easily extended to continuous mixtures:*

$$f(x) = \int_{\Omega} \pi(\omega)\, g(x;\omega)\, \mathrm{d}\omega$$

where $\omega \in \Omega$ is a parameter of the distribution $\mathbf{G}$. For instance, we have seen that the negative binomial distribution is a gamma-Poisson mixture distribution:

$$\begin{cases} \mathcal{NB}(r,p) \sim \mathcal{P}(\Lambda) \\ \Lambda \sim \mathcal{G}(r,(1-p)/p) \end{cases}$$

To simulate the negative binomial distribution, we first simulate the gamma random variate $g \sim \mathcal{G}(r,(1-p)/p)$, and then simulate the Poisson random variable, whose parameter[7] λ is equal to g.

[7] This means that the parameter λ changes at each simulation.

13.1.3 Generating random vectors

In this section, we consider algorithms for simulating a random vector $X = (X_1, \ldots, X_n)$ from a given distribution function $\mathbf{F}(x) = \mathbf{F}(x_1, \ldots, x_n)$. In fact, the previous methods used to generate a random variable are still valid in the multidimensional case.

13.1.3.1 Method of conditional distributions

The method of inversion cannot be applied in the multivariate case, because $U = \mathbf{F}(X_1, \ldots, X_n)$ is not any longer a uniform random variable. However, if $X_1, \ldots, X_n$ are independent, we have:

$$\mathbf{F}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathbf{F}_i(x_i)$$

To simulate X, we can then generate each component $X_i \sim \mathbf{F}_i$ individually, for example by applying the method of inversion. When $X_1, \ldots, X_n$ are dependent, we have:

$$\begin{aligned} \mathbf{F}(x_1, \ldots, x_n) &= \mathbf{F}_1(x_1)\, \mathbf{F}_{2|1}(x_2 \mid x_1)\, \mathbf{F}_{3|1,2}(x_3 \mid x_1, x_2) \times \cdots \times \\ &\quad \mathbf{F}_{n|1,\ldots,n-1}(x_n \mid x_1, \ldots, x_{n-1}) \\ &= \prod_{i=1}^{n} \mathbf{F}_{i|1,\ldots,i-1}(x_i \mid x_1, \ldots, x_{i-1}) \end{aligned}$$

where $\mathbf{F}_{i|1,\ldots,i-1}(x_i \mid x_1, \ldots, x_{i-1})$ is the conditional distribution of X_i given $X_1 = x_1, \ldots, X_{i-1} = x_{i-1}$. Let us denote this '*conditional*' random variable Y_i. We notice that the random variables $(Y_1, \ldots, Y_n)$ are independent. Therefore, the underlying idea of the method of conditional distributions is to transform the random vector X by a vector Y of independent random variables. We obtain the following algorithm:

1. generate x_1 from $\mathbf{F}_1(x)$ and set $i = 2$;
2. generate x_i from $\mathbf{F}_{i|1,\ldots,i-1}(x \mid x_1, \ldots, x_{i-1})$ given $X_1 = x_1, \ldots, X_{i-1} = x_{i-1}$ and set $i = i + 1$;
3. repeat step 2 until $i = n$.

$\mathbf{F}_{i|1,\ldots,i-1}(x \mid x_1, \ldots, x_{i-1})$ is a univariate distribution function, which depends on the argument x and parameters $x_1, \ldots, x_{i-1}$. To simulate it, we can therefore use the method of inversion:

$$x_i \leftarrow \mathbf{F}^{-1}_{i|1,\ldots,i-1}(u_i \mid x_1, \ldots, x_{i-1})$$

where $\mathbf{F}^{-1}_{i|1,\ldots,i-1}$ is the inverse of the conditional distribution function and u_i is a uniform random variate.

Example 141 *We consider the bivariate logistic distribution defined as:*

$$\mathbf{F}(x_1, x_2) = \left(1 + e^{-x_1} + e^{-x_2}\right)^{-1}$$

We have $\mathbf{F}_1(x_1) = \mathbf{F}(x_1, +\infty) = (1 + e^{-x_1})^{-1}$. We deduce that the conditional distribution of X_2 given $X_1 = x_1$ is:

$$\begin{aligned} \mathbf{F}_{2|1}(x_2 \mid x_1) &= \frac{\mathbf{F}(x_1, x_2)}{\mathbf{F}_1(x_1)} \\ &= \frac{1 + e^{-x_1}}{1 + e^{-x_1} + e^{-x_2}} \end{aligned}$$

We obtain:

$$\mathbf{F}_1^{-1}(u) = \ln u - \ln(1-u)$$

and:

$$\mathbf{F}_{2|1}^{-1}(u \mid x_1) = \ln u - \ln(1-u) - \ln\left(1 + e^{-x_1}\right)$$

We deduce the following algorithm:

1. generate two independent uniform random variates u_1 and u_2;
2. generate x_1 from u_1:
$$x_1 \leftarrow \ln u_1 - \ln(1-u_1)$$
3. generate x_2 from u_2 and x_1:
$$x_2 \leftarrow \ln u_2 - \ln(1-u_2) - \ln\left(1 + e^{-x_1}\right)$$

Because we have $(1 + e^{-x_1})^{-1} = u_1$, the last step can be replaced by:

3'. generate x_2 from u_2 and u_1:
$$x_2 \leftarrow \ln\left(\frac{u_1 u_2}{1-u_2}\right)$$

The method of conditional distributions can be used for simulating uniform random vectors $(U_1, \ldots, U_n)$ generated by copula functions. In this case, we have:

$$\begin{aligned}\mathbf{C}(u_1, \ldots, u_n) &= \mathbf{C}_1(u_1)\,\mathbf{C}_{2|1}(u_2 \mid u_1)\,\mathbf{C}_{3|1,2}(u_3 \mid u_1, u_2) \times \cdots \times \\ &\quad \mathbf{C}_{n|1,\ldots,n-1}(u_n \mid u_1, \ldots, u_{n-1}) \\ &= \prod_{i=1}^{n} \mathbf{C}_{i|1,\ldots,i-1}(u_i \mid u_1, \ldots, u_{i-1})\end{aligned}$$

where $\mathbf{C}_{i|1,\ldots,i-1}(u_i \mid u_1, \ldots, u_{i-1})$ is the conditional distribution of U_i given $U_1 = u_1, \ldots, U_{i-1} = u_{i-1}$ By definition, we have $\mathbf{C}_1(u_1) = u_1$. We obtain the following algorithm:

1. generate n independent uniform random variates $v_1, \ldots, v_n$;
2. generate $u_1 \leftarrow v_1$ and set $i = 2$;
3. generate u_i by finding the root of the equation:
$$\mathbf{C}_{i|1,\ldots,i-1}(u_i \mid u_1, \ldots, u_{i-1}) = v_i$$
and set $i = i + 1$;
4. repeat step 3 until $i = n$.

For some copula functions, there exists an analytical expression of the inverse of the conditional copula. In this case, the third step is replaced by:

3'. generate u_i by the inversion method:
$$u_i \leftarrow \mathbf{C}_{i|1,\ldots,i-1}^{-1}(v_i \mid u_1, \ldots, u_{i-1})$$

Remark 153 *For any probability distribution, the conditional distribution can be calculated as follows:*

$$\mathbf{F}_{i|1,\ldots,i-1}(x_i \mid x_1,\ldots,x_{i-1}) = \frac{\mathbf{F}(x_1,\ldots,x_{i-1},x_i)}{\mathbf{F}(x_1,\ldots,x_{i-1})}$$

In particular, we have:

$$\begin{aligned}\partial_1 \mathbf{F}(x_1,x_2) &= \partial_1\left(\mathbf{F}_1(x_1)\cdot\mathbf{F}_{2|1}(x_2 \mid x_1)\right)\\ &= f_1(x_1)\cdot\mathbf{F}_{2|1}(x_2 \mid x_1)\end{aligned}$$

For copula functions, the density $f_1(x_1)$ is equal to 1, meaning that:

$$\mathbf{C}_{2|1}(u_2 \mid u_1) = \partial_1 \mathbf{C}(u_1,u_2)$$

We can generalize this result and show that the conditional copula given some random variables U_i for $i \in \Omega$ is equal to the cross-derivative of the copula function $\mathbf{C}$ with respect to the arguments u_i for $i \in \Omega$.

We recall that Archimedean copulas are defined as:

$$\mathbf{C}(u_1,u_2) = \varphi^{-1}(\varphi(u_1)+\varphi(u_2))$$

where $\varphi(u)$ is the generator function. We have:

$$\varphi(\mathbf{C}(u_1,u_2)) = \varphi(u_1)+\varphi(u_2)$$

and:

$$\varphi'(\mathbf{C}(u_1,u_2))\cdot\frac{\partial\,\mathbf{C}(u_1,u_2)}{\partial\,u_1} = \varphi'(u_1)$$

We deduce the following expression of the conditional copula:

$$\begin{aligned}\mathbf{C}_{2|1}(u_2 \mid u_1) &= \frac{\partial\,\mathbf{C}(u_1,u_2)}{\partial\,u_1}\\ &= \frac{\varphi'(u_1)}{\varphi'(\varphi^{-1}(\varphi(u_1)+\varphi(u_2)))}\end{aligned}$$

The calculation of the inverse function gives:

$$\mathbf{C}_{2|1}^{-1}(v \mid u_1) = \varphi^{-1}\left(\varphi\left(\varphi'^{-1}\left(\frac{\varphi'(u_1)}{v}\right)\right)-\varphi(u_1)\right)$$

We obtain the following algorithm for simulating Archimedean copulas:

1. generate two independent uniform random variates v_1 and v_2;
2. generate $u_1 \leftarrow v_1$;
3. generate u_2 by the inversion method:

$$u_2 \leftarrow \varphi^{-1}\left(\varphi\left(\varphi'^{-1}\left(\frac{\varphi'(u_1)}{v_2}\right)\right)-\varphi(u_1)\right)$$

Example 142 *We consider the Clayton copula:*

$$\mathbf{C}(u_1,u_2) = \left(u_1^{-\theta}+u_2^{-\theta}-1\right)^{-1/\theta}$$

The Clayton copula is an Archimedean copula, whose generator function is:

$$\varphi(u) = u^{-\theta} - 1$$

We deduce that:

$$\begin{aligned}
\varphi^{-1}(u) &= (1+u)^{-1/\theta} \\
\varphi'(u) &= -\theta u^{-(\theta+1)} \\
\varphi'^{-1}(u) &= (-u/\theta)^{-1/(\theta+1)}
\end{aligned}$$

After some calculations, we obtain:

$$\mathbf{C}_{2|1}^{-1}(v \mid u_1) = \left(1 + u_1^{-\theta}\left(v^{-\theta/(\theta+1)} - 1\right)\right)^{-1//\theta}$$

In Table 13.4, we simulate five realizations of the Clayton copula using the inverse function of the conditional copula. In the case $\theta = 0.01$, u_2 is close to v_2 because the Clayton copula is the product copula $\mathbf{C}^{\perp}$ when θ tends to 0. In the case $\theta = 1.5$, we note the impact of the conditional copula on the simulation of u_2.

TABLE 13.4: Simulation of the Clayton copula

Random uniform variates		Clayton copula $\theta = 0.01$		Clayton copula $\theta = 1.5$	
v_1	v_2	u_1	u_2	u_1	u_2
0.2837	0.4351	0.2837	0.4342	0.2837	0.3296
0.0386	0.2208	0.0386	0.2134	0.0386	0.0297
0.3594	0.5902	0.3594	0.5901	0.3594	0.5123
0.3612	0.3268	0.3612	0.3267	0.3612	0.3247
0.0797	0.6479	0.0797	0.6436	0.0797	0.1704

13.1.3.2 Method of transformation

To simulate a Gaussian random vector $X \sim \mathcal{N}(\mu, \Sigma)$, we consider the following transformation:

$$X = \mu + A \cdot N$$

where $AA^{\top} = \Sigma$ and $N \sim \mathcal{N}(\mathbf{0}, I)$. Therefore, we can simulate a correlated Gaussian random vector by using n independent Gaussian random variates $\mathcal{N}(0,1)$ and finding a square matrix A such that $AA^{\top} = \Sigma$. Since we know that Σ is a positive definite symmetric matrix, it has a unique Cholesky decomposition:

$$\Sigma = PP^{\top}$$

where P is a lower triangular matrix.

Remark 154 *The decomposition $AA^{\top} = \Sigma$ is not unique. For instance, if we use the eigendecomposition:*

$$\Sigma = U\Lambda U^{\top}$$

we can set $A = U\Lambda^{1/2}$. Indeed, we have:

$$\begin{aligned}
AA^{\top} &= U\Lambda^{1/2}\Lambda^{1/2}U^{\top} \\
&= U\Lambda U^{\top} \\
&= \Sigma
\end{aligned}$$

To simulate a multivariate Student's t distribution $Y = (Y_1, \ldots, Y_n) \sim \mathbf{T}_n(\Sigma, \nu)$, we use the relationship:

$$Y_i = \frac{X_i}{\sqrt{Z/\nu}}$$

where the random vector $X = (X_1, \ldots, X_n) \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and the random variable $Z \sim \chi^2(\nu)$ are independent.

The transformation method is particularly useful for simulating copula functions. Indeed, if $X = (X_1, \ldots, X_n) \sim \mathbf{F}$, then the probability distribution of the random vector $U = (U_1, \ldots, U_n)$ defined by:

$$U_i = \mathbf{F}_i(X)$$

is the copula function $\mathbf{C}$ associated to $\mathbf{F}$.

Example 143 *To simulate the Normal copula with the matrix of parameters ρ, we simulate $N \sim \mathcal{N}(\mathbf{0}, I)$ and apply the transformation:*

$$U = \Phi(P \cdot N)$$

where P is the Cholesky decomposition of the correlation matrix ρ.

Example 144 *To simulate the Student's t copula with the matrix of parameters ρ and ν degrees of freedom, we simulate $T \sim \mathbf{T}_n(\rho, \nu)$ and apply the transformation:*

$$U_i = \mathbf{T}_v(T_i)$$

In Figures 13.5 and 13.6, we draw 1 024 simulations of Normal and t_1 copulas for different values of ρ. We notice that the Student's t copula correlates the extreme values more than the Normal copula.

On page 735, frailty copulas have been defined as:

$$\mathbf{C}(u_1, \ldots, u_n) = \psi\left(\psi^{-1}(u_1) + \ldots + \psi^{-1}(u_n)\right)$$

where $\psi(x)$ is the Laplace transform of a random variable X. Using the mixture representation of frailty copulas, Marshall and Olkin (1988) showed that they can be generated using the following algorithm:

1. simulate n independent uniform random variates $v_1, \ldots, v_n$;
2. simulate the frailty random variate x with the Laplace transform ψ;
3. apply the transformation:

$$(u_1, \ldots, u_n) \leftarrow \left(\psi\left(-\frac{\ln u_1}{x}\right), \ldots, \psi\left(-\frac{\ln u_n}{x}\right)\right)$$

For instance, the Clayton copula is a frailty copula where $\psi(x) = (1+x)^{-1/\theta}$ is the Laplace transform of the gamma random variable $\mathcal{G}(1/\theta, 1)$. Therefore, the algorithm to simulate the Clayton copula is:

$$\begin{cases} x \leftarrow \mathcal{G}(1/\theta, 1) \\ (u_1, \ldots, u_n) \leftarrow \left(\left(1 - \frac{\ln u_1}{x}\right)^{-1/\theta}, \ldots, \left(1 - \frac{\ln u_n}{x}\right)^{-1/\theta}\right) \end{cases}$$

Examples of simulating the Clayton copula using this algorithm is given in Figure 13.7.

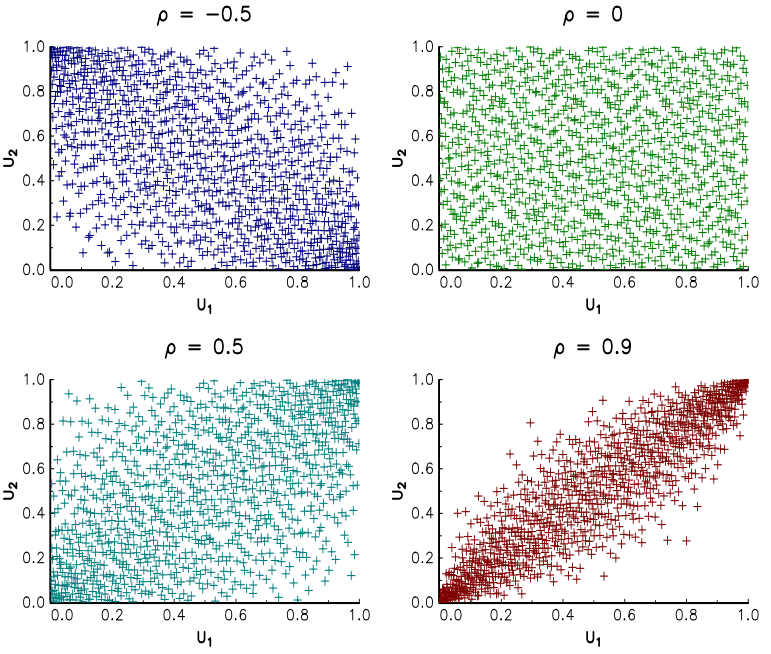

FIGURE 13.5: Simulation of the Normal copula

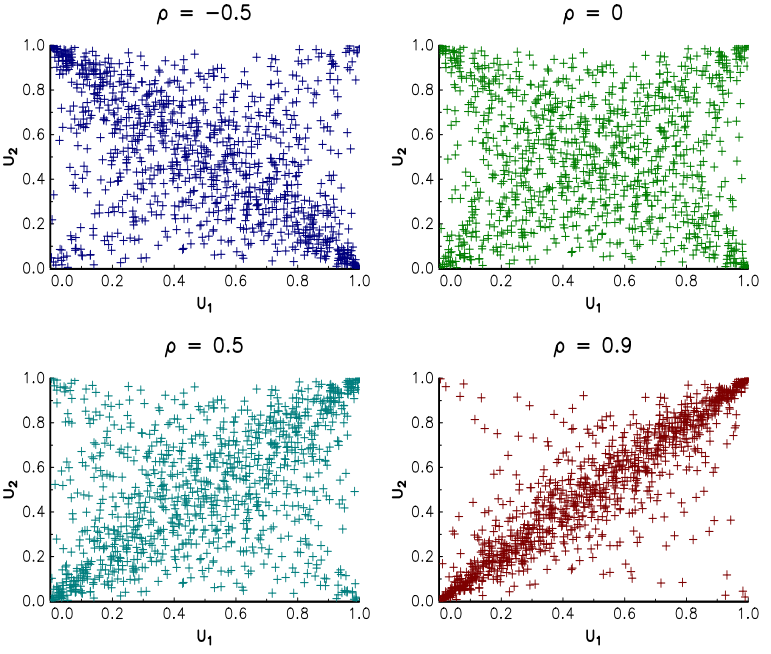

FIGURE 13.6: Simulation of the t_1 copula

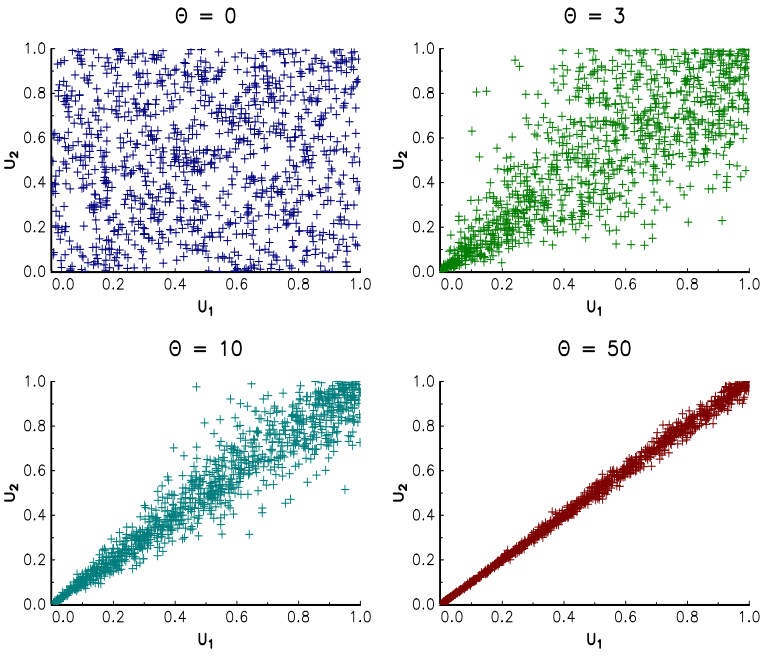

FIGURE 13.7: Simulation of the Clayton copula

Remark 155 *For other frailty copulas, the reader can refer to the survey of McNeil (2008) for the list of Laplace transforms and corresponding algorithms to simulate the frailty random variable.*

We now consider the multivariate distribution $\mathbf{F}(x_1, \ldots, x_n)$, whose canonical decomposition is defined as:

$$\mathbf{F}(x_1, \ldots, x_n) = \mathbf{C}(\mathbf{F}_1(x_1), \ldots, \mathbf{F}_n(x_n))$$

We recall that if $(U_1, \ldots, U_n) \sim \mathbf{C}$, the random vector $(X_1, \ldots, X_n) = \left(\mathbf{F}_1^{-1}(U_1), \ldots, \mathbf{F}_n^{-1}(U_n)\right)$ follows the distribution function $\mathbf{F}$. We deduce the following algorithm:

$$\begin{cases} (u_1, \ldots, u_n) \leftarrow \mathbf{C} \\ (x_1, \ldots, x_n) \leftarrow \left(\mathbf{F}_1^{-1}(u_1), \ldots, \mathbf{F}_n^{-1}(u_n)\right) \end{cases}$$

Let us consider that the default time $\boldsymbol{\tau}$ and the loss given default LGD of one counterparty are distributed according to the exponential distribution $\mathcal{E}(5\%)$ and the beta distribution $\mathcal{B}(2, 2)$. We also assume that the default time and the loss given default are correlated and the dependence function is a Clayton copula. In Figure 13.8, we use the Clayton random variates generated in Figure 13.7 and apply exponential and beta inverse transforms to them. For the beta distribution, we use the Newton-Raphson algorithm to generate the LGD random variable.

The previous algorithms suppose that we know the analytical expression $\mathbf{F}_i$ of the univariate probability distributions in order to calculate the quantile function $\mathbf{F}_i^{-1}$. This is not always the case. For instance, in the operational risk, the loss of the bank is equal to the sum of aggregate losses:

$$L = \sum_{k=1}^{K} S_k$$

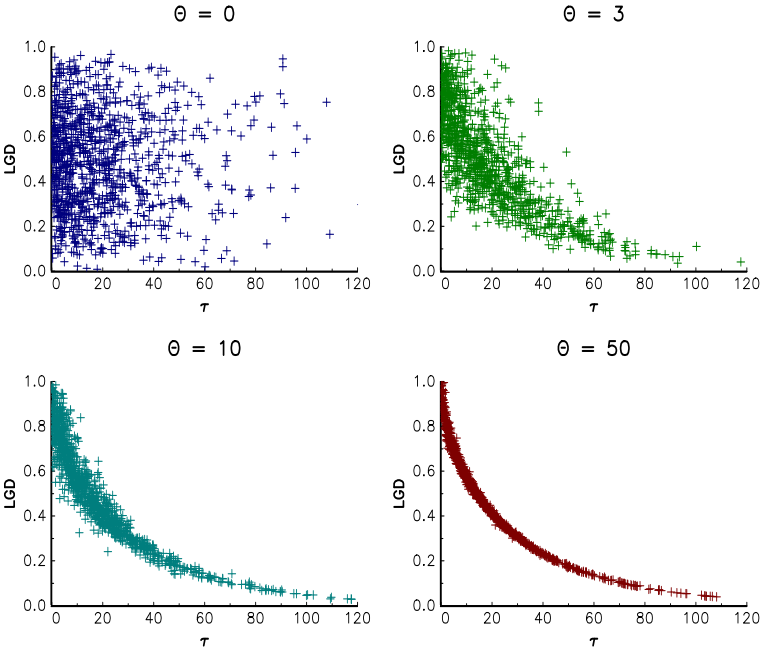

FIGURE 13.8: Simulation of the correlated random vector $(\boldsymbol{\tau}, \text{LGD})$

where S_k is also the sum of individual losses for the k^{th} cell of the mapping matrix. In practice, the probability distribution of S_k is estimated by the method of simulations. In this case, we have to use the method of the empirical quantile function. Let $\mathbb{F}_{i,m}$ be the empirical process of X_i. We know that:

$$\sup_x |\mathbb{F}_{i,m}(x) - \mathbf{F}_i(x)| \to 0 \qquad \text{when } m \to \infty$$

We note $\mathbb{U}_m$ and $\mathbb{F}_m$ the empirical processes corresponding to the distribution functions $\mathbf{C}(u_1, \ldots, u_n)$ and $\mathbf{F}(x_1, \ldots, x_n)$. The Glivenko-Cantelli theorem tells us that:

$$\sup_{x_1,\ldots,x_n} |\mathbb{F}_m(x_1, \ldots, x_n) - \mathbf{F}(x_1, \ldots, x_n)| \to 0 \qquad \text{when } m \to \infty$$

We deduce that:

$$\sup_{u_1,\ldots,u_n} \left|\mathbb{U}_{m_2}\left(\mathbb{F}_{1,m_1}^{-1}(u_1), \ldots, \mathbb{F}_{n,m_1}^{-1}(u_n)\right) - \mathbf{C}\left(\mathbf{F}_1^{-1}(u_1), \ldots, \mathbf{F}_n^{-1}(u_n)\right)\right| \to 0$$

when both m_1 and m_2 tend to ∞. It follows that the method of the empirical quantile function is implemented as follows:

1. for each random variable X_i, simulate m_1 random variates $x_{i,m}^\star$ and estimate the empirical distribution $\hat{\mathbf{F}}_i$;
2. simulate a random vector $(u_1, \ldots, u_n)$ from the copula function $\mathbf{C}(u_1, \ldots, u_n)$;
3. simulate the random vector $(x_1, \ldots, x_n)$ by inverting the empirical distributions $\hat{\mathbf{F}}_i$:
$$x_i \leftarrow \hat{\mathbf{F}}_i^{-1}(u_i)$$
we also have:
$$x_i \leftarrow \inf\left\{x \,\middle|\, \frac{1}{m_1}\sum_{m=1}^{m_1} \mathbf{1}\left\{x \le x_{i,m}^\star\right\} \ge u_i\right\}$$
4. repeat steps 2 and 3 m_2 times.

In Figure 13.9, we illustrate this algorithm by assuming that $X_1 \sim \mathcal{N}(0,1)$, $X_2 \sim \mathcal{N}(0,1)$ and the dependence function of (X_1, X_2) is the Clayton copula with parameter $\theta = 3$. If we use $m_1 = 50$ simulations to estimate the quantile function of X_1 and X_2, the approximation is not good. However, when we consider a large number of simulations ($m_1 = 5000$), we obtain simulated values of the random vector (X_1, X_2) that are close to the simulated values calculated with the analytical quantile function $\Phi^{-1}(u)$. We now consider a more complex example. We assume that $X_1 \sim \mathcal{N}(-1,2)$, $X_2 \sim \mathcal{N}(0,1)$, $Y_1 \sim \mathcal{G}(0.5)$ and $Y_2 \sim \mathcal{G}(1,2)$ are four independent random variables. Let $(Z_1 = X_1 + Y_1, Z_2 = X_2 \cdot Y_2)$ be the random vector, whose dependence function is the t copula with parameters $\nu = 2$ and $\rho = -70\%$. It is obvious that it is not possible to find an analytical expression of the marginal distributions of Z_1 and Z_2. However, the random variables Z_1 and Z_2 are easy to simulate (Figure 13.10). This is why we can use the method of the empirical quantile function to simulate the random vector (Z_1, Z_2). A sample of 4 000 simulated values of the vector (Z_1, Z_2) is reported in Figure 13.11.

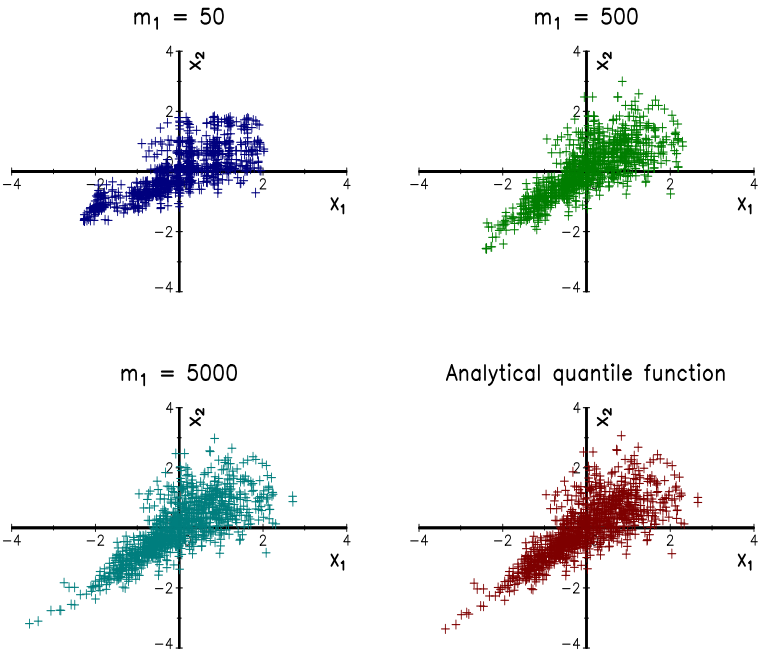

FIGURE 13.9: Convergence of the method of the empirical quantile function

13.1.4 Generating random matrices

The simulation of random matrices is a specialized topic, which is generally not covered by textbooks. However, the tools presented in this section are very useful in finance. This is particularly true when we would like to measure the correlation risk.

13.1.4.1 Orthogonal and covariance matrices

An orthogonal matrix Q is a square $n \times n$ matrix, whose columns and rows are orthonormal vectors:

$$Q^\top Q = QQ^\top = I_n$$

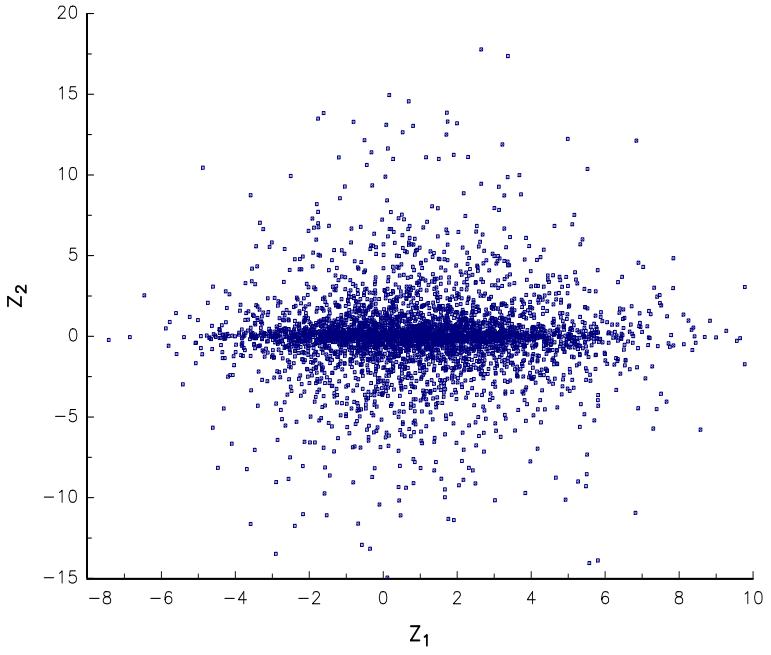

FIGURE 13.10: Simulation of the random variables Z_1 and Z_2

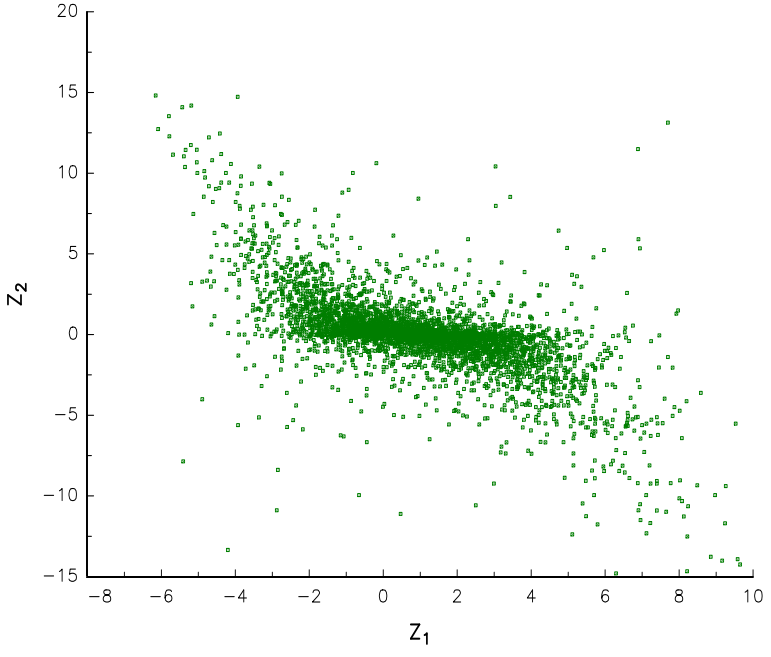

FIGURE 13.11: Simulation of the random vector (Z_1, Z_2)

It follows that $Q^{-1} = Q$. Generally, Monte Carlo methods require generation of random orthogonal matrices distributed according to the Haar measure[8]. Anderson *et al.* (1987) proposed two simple algorithms to generate Q:

1. Let X be a $n \times n$ matrix of independent standard Gaussian random variables. Q is the unitary matrix of the QR factorization of $X = QR$ where R is an upper triangular factorization.

2. Let X be a $n \times p$ matrix of independent standard Gaussian random variables with $p \geq n$. Q corresponds to the matrix V of the eigendecomposition $X^\top X = V \Lambda V^\top$ or the matrix U of the singular value decomposition $X^\top X = U \Sigma V^*$.

Stewart (1980) proposed another popular algorithm based on the Household transformation. Let H_x be the symmetric orthogonal matrix defined as:

$$H_x x = \pm \|x\| \cdot \mathbf{e}_1$$

We consider a series of independent Gaussian random vectors: $x_1 \sim \mathcal{N}_n(0, I_n)$, $x_2 \sim \mathcal{N}_{n-1}(0, I_{n-1})$, etc. We form the matrix $\tilde{H}_k = \text{diag}(I_{k-1}, H_{x_k})$. The random orthogonal matrix Q is then generated by the product:

$$Q = \left(\prod_{k=1}^{n-1} \tilde{H}_k \right) D$$

where D is the diagonal matrix with entries ± 1. To illustrate this algorithm, we simulate random orthogonal matrices Q for different values of n, and we report the distribution of the eigenvalues of Q in Figure 13.12. We verify that they are almost uniformly distributed on the unit sphere.

Remark 156 *To simulate a random covariance matrix Σ with specified eigenvalues $\lambda_1, \ldots, \lambda_n$, we generate a random orthogonal matrix Q and consider the transformation:*

$$\Sigma = Q \Lambda Q^\top$$

where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$.

13.1.4.2 Correlation matrices

A correlation matrix $\mathbb{C}$ is a symmetric positive definite matrix, whose diagonal elements are equal to 1. It follows that the sum of the eigenvalues is exactly equal to n. The previous algorithm can be used to simulate a random correlation matrix. Indeed, we only need to transform Σ into $\mathbb{C}$:

$$\mathbb{C}_{i,j} = \frac{\Sigma_{i,j}}{\sqrt{\Sigma_{i,i} \cdot \Sigma_{j,j}}}$$

However, this method is not always interesting, because it does not preserve the specified eigenvalues $\lambda_1, \ldots, \lambda_n$. Let us consider an example with $\lambda_1 = 0.5$, $\lambda_2 = 1.00$ and $\lambda_3 = 1$. A simulation of Σ gives:

$$\Sigma = \begin{pmatrix} 1.28570 & -0.12868 & 0.37952 \\ -0.12868 & 0.89418 & 0.16377 \\ 0.37952 & 0.16377 & 0.82012 \end{pmatrix}$$

[8] Any column or any row of Q has a uniform distribution over the n-dimensional unit sphere.

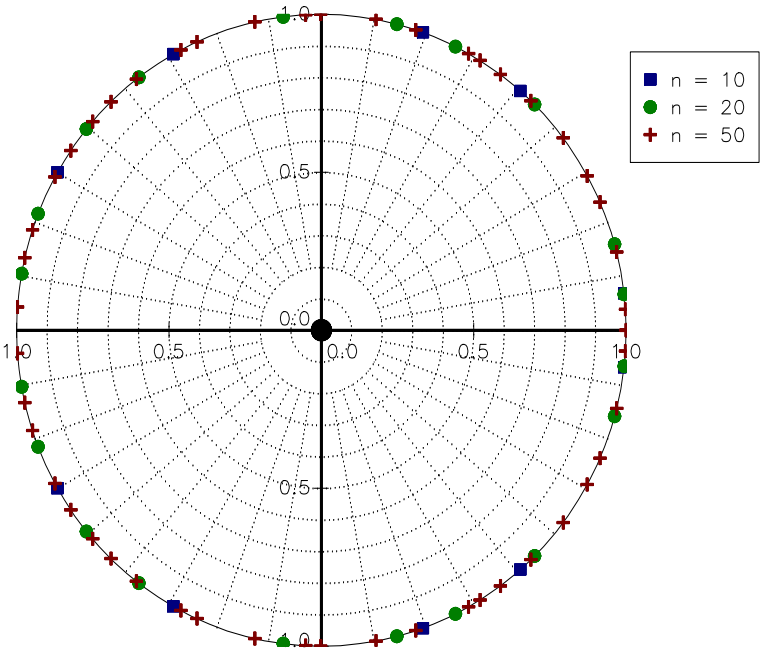

FIGURE 13.12: Distribution of the eigenvalues of simulated random orthogonal matrices

We deduce the following random correlation matrix:

$$\mathbb{C} = \begin{pmatrix} 1.00000 & -0.12001 & 0.36959 \\ -0.12001 & 1.00000 & 0.19124 \\ 0.36959 & 0.19124 & 1.00000 \end{pmatrix}$$

If we calculate the eigenvalues of $\mathbb{C}$, we obtain $\lambda_1 = 1.378$, $\lambda_2 = 1.095$ and $\lambda_3 = 0.527$. The problem comes from the fact that $Q\Lambda Q^\top$ generates a covariance matrix with specified eigenvalues, but never a correlation matrix even if the sum of the eigenvalues is equal to n.

Bendel and Mickey (1978) proposed an algorithm to transform the matrix Σ into a correlation matrix $\mathbb{C}$ with specified eigenvalues $\lambda_1, \ldots, \lambda_n$. The main idea is to perform Givens rotations[9]. Let $G_{c,s}(i,j)$ be the Givens matrix:

$$G_{c,s}(i,j) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & & & & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & & \ddots & & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & & & & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

such that the (i,j) element[10] of $G_{c,s}(i,j)^\top \Sigma G_{c,s}(i,j)$ is equal to 1. By performing n successive Givens transformations $\Sigma \leftarrow G_{c,s}(i,j)^\top \Sigma G_{c,s}(i,j)$, we obtain a correlation matrix $\mathbb{C}$

[9] A Givens rotation is a rotation in the plane spanned by two coordinates axes (Golub and Van Loan, 2013). Because Givens matrices are orthogonal, eigenvalues are not changed.

[10] We have $i < j$ and $\Sigma_{i,i} < 1 < \Sigma_{j,j}$ (or $\Sigma_{i,i} > 1 > \Sigma_{j,j}$).

with eigenvalues $\lambda_1, \ldots, \lambda_n$. The previous algorithm has been extensively studied by Davies and Higham (2000), who showed that:

$$c = \frac{1}{\sqrt{1+t^2}} \text{ and } s = c \cdot t$$

where:

$$t = \frac{\Sigma_{i,j} + \sqrt{\Sigma_{i,j}^2 - (\Sigma_{i,i} - 1)(\Sigma_{j,j} - 1)}}{(\Sigma_{j,j} - 1)}$$

To show the difference between the Bendel-Mickey algorithm and the previous covariance algorithm, we simulate a correlation matrix of dimension 20 with specified eigenvalues and the two algorithms. In Figure 13.13, we compare the eigenvalues calculated with the simulated correlation matrices and compare them with the specified eigenvalues. We verify that the Bendel-Mickey algorithm preserves the spectrum, which is not the case of the covariance algorithm[11].

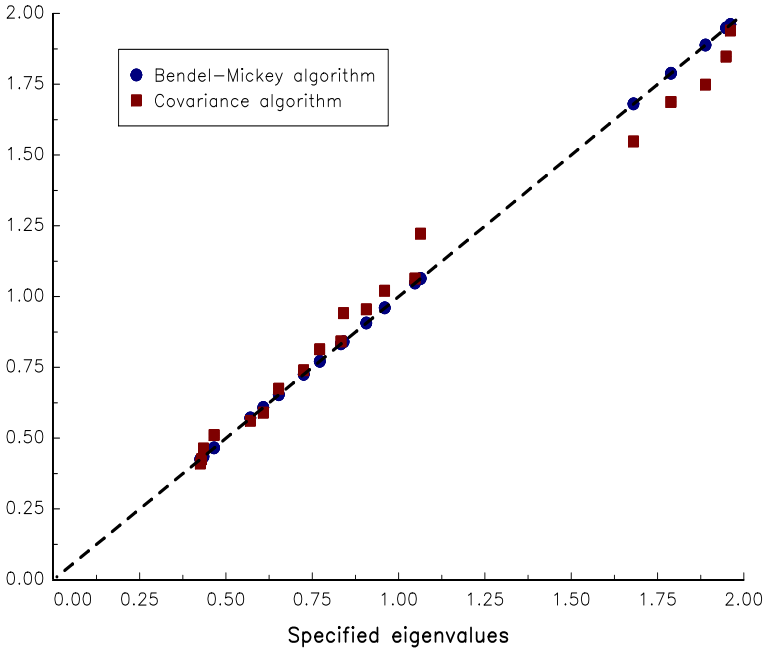

FIGURE 13.13: Comparison of the Bendel-Mickey and covariance algorithms

We study the correlation risk of the basket option, whose payoff is equal to:

$$G = (S_1(T) - S_2(T) + S_3(T) - S_4(T) - K)_+ \cdot \mathbb{1}\{S_5(T) > L\}$$

where $S_i(T)$ is the price of the i^{th} asset at the maturity T. We assume that the dynamics of the asset prices follows a Black-Scholes model:

$$S_i(T) = S_i(0) \cdot \exp\left(\left(r - \frac{1}{2}\sigma_i^2\right)T + \sigma_i(W_i(T) - W_i(0))\right)$$

[11] However, we notice that the eigenvalues are close.

where r is the risk-free rate, σ_i is the asset volatility and W_i is a Brownian process. We also assume that the Brownian processes are correlated:

$$\mathbb{E}\left[W_i\left(t\right)W_j\left(t\right)\right] = \rho_{i,j}\, t$$

To calculate the price of the basket option, we simulate the terminal value of $S_i\left(T\right)$ and average the simulated payoff G_s:

$$P = \mathbb{E}\left[e^{-rT}G\right] \approx \frac{1}{n_S}\sum_{s=1}^{n_S} e^{-rT}G_s$$

where n_S is the number of simulations. We use the following values: $S_i\left(0\right) = 100$, $r = 5\%$, $\sigma_i = 20\%$, $T = 0.25$, $K = 5$ and $L = 105$. We consider that it is difficult to estimate the correlation matrix and assume that it is unstable. In this case, we have to find an upper bound for P in order to take into account this correlation risk. Generally, we price the option by using a constant correlation matrix $\mathbb{C}_5\left(\rho\right)$ and takes the supremum:

$$P^+ = \sup_{\rho} P\left(\mathbb{C}_5\left(\rho\right)\right)$$

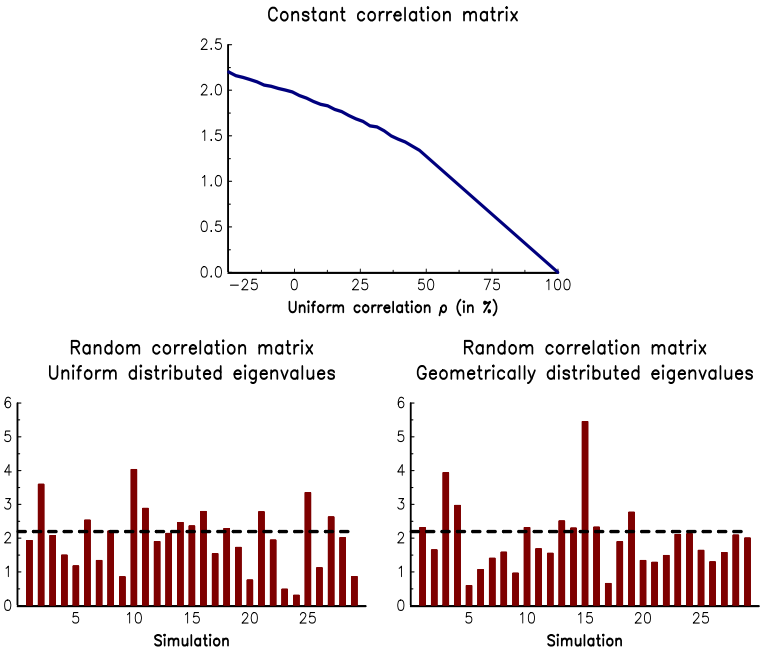

FIGURE 13.14: Price of the basket option

In the top panel in Figure 13.14, we report the price of the basket option with respect to the uniform correlation ρ. We notice that the price is a decreasing function of ρ and reaches its maximum when the uniform correlation is -25%. Therefore, we could suppose that the upper bound is equal to \$2.20. However, if we consider random correlation matrices, we observe that this price is not conservative (see bottom panels in Figure 13.14). For instance,

we obtain a price equal to \$5.45 with the following correlation matrix:

$$\mathbb{C} = \begin{pmatrix} 1.0000 & -0.4682 & -0.3034 & -0.1774 & 0.1602 \\ -0.4682 & 1.0000 & -0.3297 & 0.1381 & -0.7272 \\ -0.3034 & -0.3297 & 1.0000 & -0.3273 & 0.6106 \\ -0.1774 & 0.1381 & -0.3273 & 1.0000 & -0.1442 \\ 0.1602 & -0.7272 & 0.6106 & -0.1442 & 1.0000 \end{pmatrix}$$

This matrix indicates the type of correlation risks we face when we want to hedge this basket option. Indeed, the correlation risk is maximum when the fifth asset (which activates the barrier) is positively correlated with the first and third assets and negatively correlated with the second and fourth assets.

In the Bendel-Mickey algorithm, we control the structure of the random correlation matrix by specifying the eigenvalues. In Finance, it can be not sufficient. For instance, we may want to simulate the matrix $\mathbb{C}$ such that is expected value is equal to a given correlation matrix $\mathbb{C}^\star$:

$$\mathbb{E}[\mathbb{C}] = \mathbb{C}^\star$$

Let A be a random symmetric matrix with zeros on the diagonal and mean $\mathbb{E}[A] = \mathbf{0}$. Marsaglia and Olkin (1984) showed that $\mathbb{C} = A + \mathbb{C}^\star$ is a random correlation matrix with $\mathbb{E}[A + \mathbb{C}^\star] = \mathbb{C}^\star$ if the 2-norm of A is less than the smallest eigenvalue $\lambda_{\min}$ of $\mathbb{C}^\star$. There is a variety of algorithms that uses this result. For instance, Marsaglia and Olkin (1984) proposed to generate a random correlation matrix R with specified eigenvalues in the interval $[1 - \lambda_{\min}, 1 + \lambda_{\min}]$ and to take $\mathbb{C} = (R - I_n) + \mathbb{C}^\star$.

13.1.4.3 Wishart matrices

To generate a random Wishart matrix S, we simulate n independent Gaussian random vectors $X_i \sim \mathcal{N}_p(0, \Sigma)$ and form the $n \times p$ matrix X by concatenating the random vectors in the following way:

$$X = \begin{pmatrix} X_1^\top \\ \vdots \\ X_n^\top \end{pmatrix}$$

Then, we have $S = X^\top X$. The simulation of an inverse Wishart matrix T is straightforward by applying the transformation method:

$$T = S^{-1}$$

13.2 Simulation of stochastic processes

We distinguish two types of time series models, those based on discrete-time stochastic processes and those based on continuous-time stochastic processes. Discrete-time models are easier to simulate, in particular when we consider time-homogeneous Markov processes. This is not the case of continuous-time models, which are generally approximated by a time-discretized process. In this case, the convergence of the discrete simulation to the continuous solution depends on the approximation scheme.

13.2.1 Discrete-time stochastic processes

13.2.1.1 Correlated Markov chains

We consider a vector $\mathfrak{R} = (\mathfrak{R}_1, \ldots, \mathfrak{R}_M)$ of time-homogeneous Markov chains, whose transition probability matrix is P. The simulation of the Markov chain $\mathfrak{R}_m$ is given by the following algorithm:

1. we assume the initial position of the Markov chain:

$$\mathfrak{R}_m(0) = i_0$$

2. let u be a random number; we simulate the new position of $\mathfrak{R}_m$ by inverting the conditional probability distribution, whose elements are:

$$\Pr(\mathfrak{R}_m(n+1) = i_{n+1} \mid \mathfrak{R}_m(n) = i_n) = p_{i_n,i_{n+1}} = e_{i_n}^\top P e_{i_{n+1}}$$

   we have:

$$i_{n+1} = \left\{ k : \sum_{j=1}^{k-1} p_{i_n,j} < u \leq \sum_{j=1}^{k} p_{i_n,j} \right\}$$

3. we go back to step 2.

We now assume that the dependence between the Markov chains $(\mathfrak{R}_1, \ldots, \mathfrak{R}_M)$ is given by a copula function $\mathbf{C}$, implying that the Markov chains are correlated. The algorithm becomes:

1. we assume the initial position of the Markov chains:

$$\mathfrak{R}_m(0) = i_{m,0}$$

2. let $(u_1, \ldots, u_M)$ be a vector of correlated uniform random numbers such that:

$$(u_1, \ldots, u_M) \sim \mathbf{C}$$

3. for each Markov chain m, we simulate the new position of $\mathfrak{R}_m$ by inverting the conditional probability distribution; we have:

$$i_{m,n+1} = \left\{ k : \sum_{j=1}^{k-1} p_{i_{m,n},j} < u_m \leq \sum_{j=1}^{k} p_{i_{m,n},j} \right\}$$

   and $\mathfrak{R}_m(n+1) = i_{m,n+1}$.

4. we go back to step 2.

We consider four corporate firms, whose initial credit rating is AAA, BBB, B and CCC. We assume that the rating of each company is a Markov chain $\mathfrak{R}_m$ described by the credit migration matrix given on page 208. We also assume that the dependence of the credit ratings $(\mathfrak{R}_1, \mathfrak{R}_2, \mathfrak{R}_3, \mathfrak{R}_4)$ is a Normal copula with the following matrix of parameters:

$$\rho_1 = \begin{pmatrix} 1.00 & & & \\ 0.25 & 1.00 & & \\ 0.75 & 0.50 & 1.00 & \\ 0.50 & 0.25 & 0.75 & 1.00 \end{pmatrix}$$

In Figures 13.15 and 13.17, we report 10 simulated paths of the ratings for the next 30 years. We verify that the default rating is an absorbing state. Suppose now that the parameter matrix of the Normal copula is equal to:

$$\rho_2 = \begin{pmatrix} 1.00 & & & \\ -0.25 & 1.00 & & \\ -0.75 & 0.50 & 1.00 & \\ -0.50 & 0.25 & 0.75 & 1.00 \end{pmatrix}$$

Using this correlation matrix ρ_2 instead of the previous matrix ρ_1, we obtain the results given in Figures 13.16 and 13.18. If we compare Figures 13.15 and 13.16 (or Figures 13.17 and 13.18), which are based on the same uniform random numbers, we notice that the simulated paths are not the same. The reason comes from the negative correlation between the credit rating of the first company and the other credit ratings.

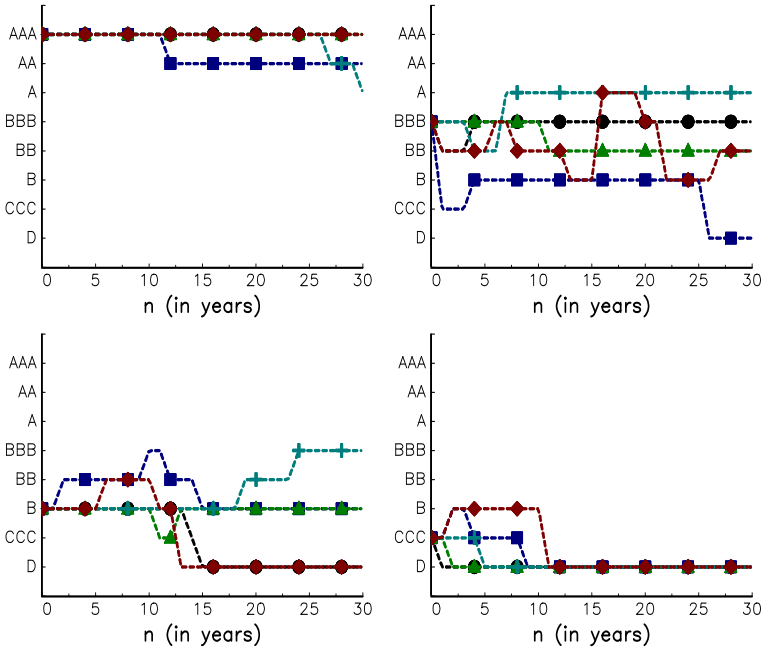

FIGURE 13.15: Simulation of rating dynamics (correlation matrix ρ_1)

13.2.1.2 Time series

A state space model (SSM) is defined by a measurement equation and a transition equation. The measurement equation describes the relationship between the observed variables y_t and the state vector α_t:

$$y_t = Z_t \alpha_t + d_t + \varepsilon_t$$

whereas the transition equation gives the dynamics of the state variables:

$$\alpha_t = T_t \alpha_{t-1} + c_t + R_t \eta_t$$

The dimension of the vectors y_t and α_t is respectively $n \times 1$ and $m \times 1$. Z_t is a $n \times m$ matrix, d_t is a $n \times 1$ vector, T_t is a $m \times m$ matrix, c_t is a $m \times 1$ vector and R_t is a $m \times p$ matrix. $\varepsilon_t \sim \mathcal{N}_n(\mathbf{0}, H_t)$ and $\eta_t \sim \mathcal{N}_p(\mathbf{0}, Q_t)$ are two independent white noise processes. By

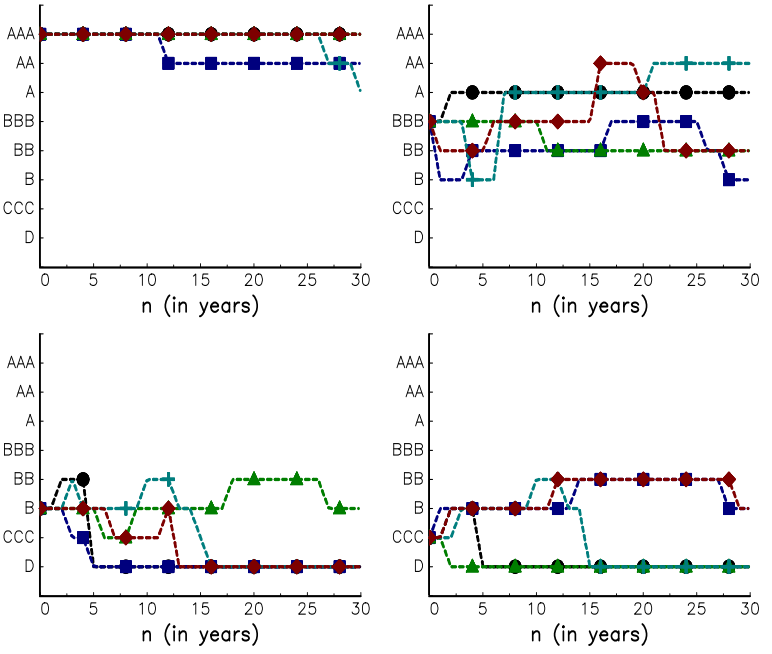

FIGURE 13.16: Simulation of rating dynamics (correlation matrix ρ_2)

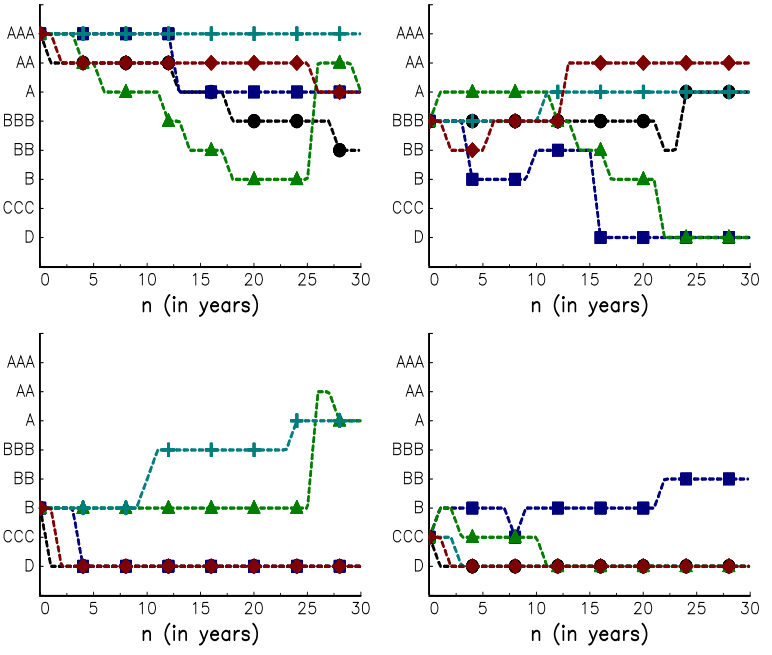

FIGURE 13.17: Simulation of rating dynamics (correlation matrix ρ_1)

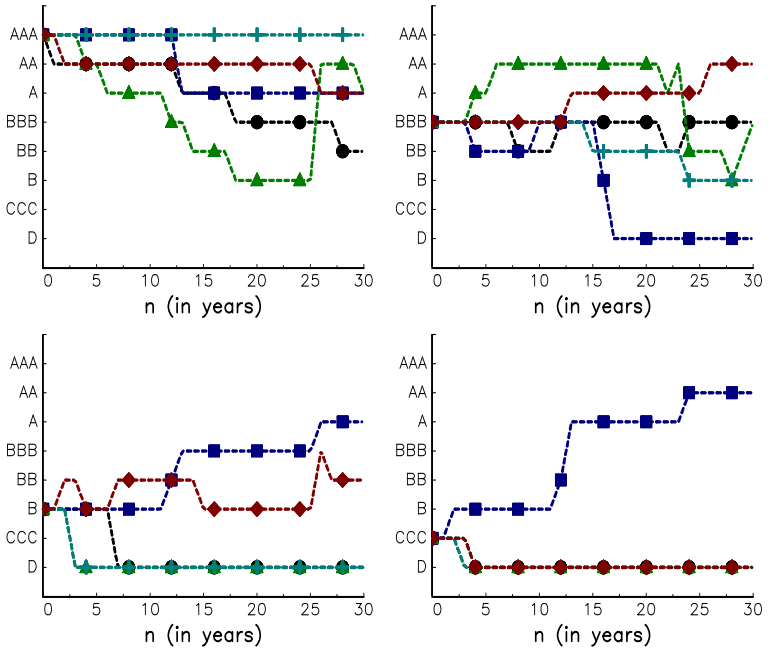

FIGURE 13.18: Simulation of rating dynamics (correlation matrix ρ_2)

construction, there is no special issue to simulate the Markov process α_t if we assume that the initial position is $\alpha_0 \sim \mathcal{N}_m(a_0, P_0)$. Indeed, we obtain the following algorithm:

1. we simulate the initial position:
$$\alpha_0 \sim \mathcal{N}_m(a_0, P_0)$$
2. we simulate the position of the state variable at time t:
$$\alpha_t \sim \mathcal{N}_m\left(T_t\alpha_{t-1} + c_t, R_tQ_tR_t^\top\right)$$
3. we simulate the space variable at time t:
$$y_t \sim \mathcal{N}_n(Z_t\alpha_t + d_t, H_t)$$
4. we go back to step 2.

Most of discrete-time stochastic processes are homogeneous, meaning that the parameters of the state space model are time-independent:

$$\begin{cases} y_t = Z\alpha_t + d + \varepsilon_t \\ \alpha_t = T\alpha_{t-1} + c + R\eta_t \end{cases}$$

where $\varepsilon_t \sim \mathcal{N}_n(\mathbf{0}, H)$ and $\eta_t \sim \mathcal{N}_p(\mathbf{0}, Q)$. In this case, the stationary solution of the transition equation is $\alpha^\star \sim \mathcal{N}_m(a^\star, P^\star)$ where $a^\star$ is equal to $(I_m - T)^{-1} c$ and $P^\star$ satisfies the matrix equation[12]:

$$P^\star = TP^\star T^\top + RQR^\top$$

[12]We also have (Harvey, 1990):

$$\operatorname{vec}(P^\star) = (I_{m^2} - T \otimes T)^{-1} \operatorname{vec}\left(RQR^\top\right)$$

In practice, we generally use the stationary solution to initialize the state space model: $\alpha_0 \sim \mathcal{N}_m(a^\star, P^\star)$.

State space models can be used to simulate structural models, AR and MA processes, vector error correction models, VAR processes, etc. For instance, a VARMA(p,q) model with K endogenous variables is defined by:

$$\left(I - \sum_{i=1}^{p} \Phi_i L^i\right) y_t = \left(I - \sum_{i=1}^{q} \Theta_i L^i\right) u_t$$

where u_t is a multidimensional white noise process. Let α_t be the vector process $(y_t, \ldots, y_{t-p+1}, u_t, \ldots, u_{t-q+1})$, whose dimension is $K(p+q)$. We have:

$$\alpha_t = T\alpha_{t-1} + Ru_t$$

where R is the $K(p+q) \times K$ matrix $\begin{bmatrix} I_K & \mathbf{0} & \cdots & \mathbf{0} & I_K & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}^\top$ and T is the $K(p+q) \times K(p+q)$ matrix:

$$T = \begin{bmatrix}
\Phi_1 & \cdots & \Phi_{p-1} & \Phi_p & \Theta_1 & \cdots & \Theta_{q-1} & \Theta_q \\
I_K & & \mathbf{0} & \mathbf{0} & \mathbf{0} & & & \\
 & \ddots & & \vdots & \vdots & & \mathbf{0} & \\
\mathbf{0} & \cdots & I_K & \mathbf{0} & \mathbf{0} & & & \\
\mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\
 & & & \mathbf{0} & I_K & & \mathbf{0} & \mathbf{0} \\
 & \mathbf{0} & & \vdots & & \ddots & & \vdots \\
 & & & \mathbf{0} & \mathbf{0} & & I_K & \mathbf{0}
\end{bmatrix}$$

We notice that:

$$y_t = Z\alpha_t$$

where Z is the $K \times K(p+q)$ matrix $\begin{bmatrix} I_K & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$. We finally obtain the following SSM representation[13]:

$$\begin{cases} y_t = Z\alpha_t \\ \alpha_t = T\alpha_{t-1} + Ru_t \end{cases}$$

13.2.2 Univariate continuous-time processes

13.2.2.1 Brownian motion

A Brownian motion (or a Wiener process) is a stochastic process $W(t)$, whose increments are stationary and independent:

$$W(t) - W(s) \sim \mathcal{N}(0, t-s)$$

Therefore, we have:

$$\begin{cases} W(0) = 0 \\ W(t) = W(s) + \epsilon(s,t) \end{cases}$$

where $\epsilon(s,t) \sim \mathcal{N}(0, t-s)$ are *iid* random variables. This representation is helpful to simulate $W(t)$ at different dates $t_1, t_2, \ldots$ If we note W_m the numerical realization of $W(t_m)$, we have:

$$W_{m+1} = W_m + \sqrt{t_{m+1} - t_m} \cdot \varepsilon_m$$

where $\varepsilon_m \sim \mathcal{N}(0,1)$ are *iid* random variables. In the case of fixed-interval times $t_{m+1} - t_m = h$, we obtain the recursion:

$$W_{m+1} = W_m + \sqrt{h} \cdot \varepsilon_m$$

[13] We have $H = \mathbf{0}$, $Q = \text{var}(u_t)$, $d = \mathbf{0}$ and $c = \mathbf{0}$.

13.2.2.2 Geometric Brownian motion

The geometric Brownian motion is described by the following stochastic differential equation:

$$\begin{cases} \mathrm{d}X(t) = \mu X(t)\,\mathrm{d}t + \sigma X(t)\,\mathrm{d}W(t) \\ X(0) = x_0 \end{cases}$$

Its solution is given by:

$$\begin{aligned} X(t) &= x_0 \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W(t)\right) \\ &= g(W(t)) \end{aligned}$$

Therefore, simulating the geometric Brownian motion $X(t)$ can be done by applying the transform method to the process $W(t)$.

Another approach to simulate $X(t)$ consists in using the following formula:

$$X(t) = X(s) \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)(t-s) + \sigma\left(W(t) - W(s)\right)\right)$$

We have:

$$X_{m+1} = X_m \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)(t_{m+1} - t_m) + \sigma\sqrt{t_{m+1} - t_m} \cdot \varepsilon_m\right)$$

where $X_m = X(t_m)$ and $\varepsilon_m \sim \mathcal{N}(0,1)$ are *iid* random variables. If we consider fixed-interval times, the numerical realization becomes:

$$X_{m+1} = X_m \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)h + \sigma\sqrt{h} \cdot \varepsilon_m\right) \tag{13.1}$$

Example 145 *In Figure 13.19, we simulate* 10 *paths of the geometric Brownian motion when* μ *and* σ *are equal to* 10% *and* 20%*. We consider a period of one year with a financial calendar of 260 trading days. This means that we use a fixed-interval time with* $h = 1/260$*. In finance, we use the convention that* $t = 1$ *corresponds to one year, which implies that* μ *and* σ *are respectively the annual expected return and volatility.*

13.2.2.3 Ornstein-Uhlenbeck process

The stochastic differential equation of the Ornstein-Uhlenbeck process is:

$$\begin{cases} \mathrm{d}X(t) = a(b - X(t))\,\mathrm{d}t + \sigma\,\mathrm{d}W(t) \\ X(0) = x_0 \end{cases}$$

We can show that the solution of the SDE is:

$$X(t) = x_0 e^{-at} + b\left(1 - e^{-at}\right) + \sigma\int_0^t e^{a(\theta - t)}\,\mathrm{d}W(\theta)$$

We also have:

$$X(t) = X(s)\,e^{-a(t-s)} + b\left(1 - e^{-a(t-s)}\right) + \sigma\int_s^t e^{a(\theta - t)}\,\mathrm{d}W(\theta)$$

where $\int_s^t e^{a(\theta-t)}\,\mathrm{d}W(\theta)$ is a Gaussian white noise process with variance $\left(1 - e^{-2a(t-s)}\right)/(2a)$. If we consider fixed-interval times, we obtain the following simulation scheme:

$$X_{m+1} = X_m e^{-ah} + b\left(1 - e^{-ah}\right) + \sigma\sqrt{\frac{1 - e^{-2ah}}{2a}} \cdot \varepsilon_m$$

where $\varepsilon_m \sim \mathcal{N}(0,1)$ are *iid* random variables.

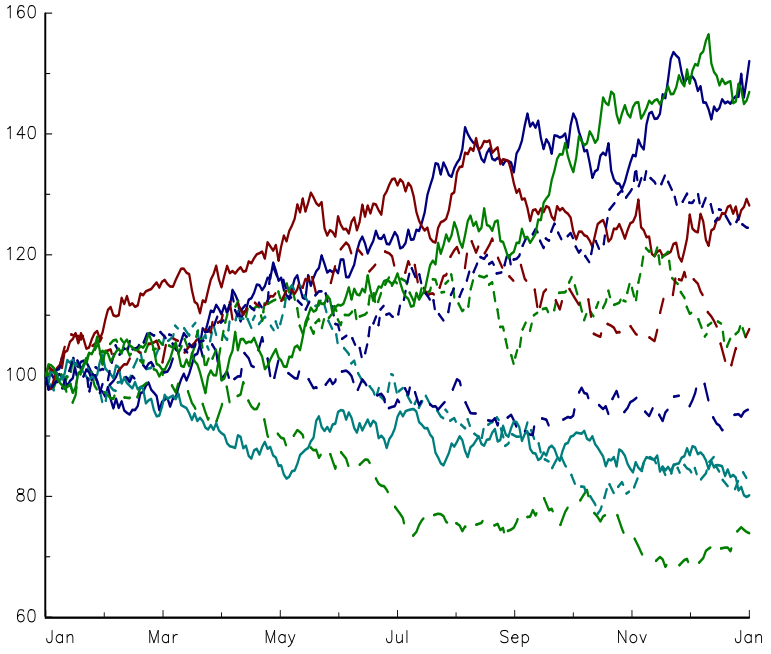

FIGURE 13.19: Simulation of the geometric Brownian motion

Example 146 *We assume that* $a = 2$, $b = 10\%$ *and* $\sigma = 1.5\%$. *The initial position of the process is* $x_0 = 5\%$. *We simulate* X_t *for two years and report the generated paths in Figure 13.20.*

13.2.2.4 Stochastic differential equations without an explicit solution

In the case of the geometric Brownian motion or the Ornstein-Uhlenbeck process, we obtain an exact scheme for simulating these processes, because we know the analytical solution. In many cases, the solution is not known and can only be simulated using approximation schemes. Let $X(t)$ be the solution of the following SDE:

$$\begin{cases} \mathrm{d}X(t) = \mu(t, X)\,\mathrm{d}t + \sigma(t, X)\,\mathrm{d}W(t) \\ X(0) = x_0 \end{cases}$$

The simplest numerical method for simulating $X(t)$ is the Euler-Maruyama scheme, which uses the following approximation:

$$X(t) - X(s) \approx \mu(t, X(s)) \cdot (t - s) + \sigma(t, X(s)) \cdot (W(t) - W(s))$$

If we consider fixed-interval times, the Euler-Maruyama scheme becomes:

$$X_{m+1} = X_m + \mu(t_m, X_m)\, h + \sigma(t_m, X_m)\, \sqrt{h} \cdot \varepsilon_m$$

where $\varepsilon_m \sim \mathcal{N}(0, 1)$ are *iid* random variables.

Remark 157 *The accuracy of numerical approximations is evaluated with the strong order of convergence. Let* $X_m^{(h)}$ *be the numerical solution of* $X(t_m)$ *computed with the constant stepwise* h. *A numerical scheme is said to converge strongly to the exact solution if we have:*

$$\lim_{h \to 0} \mathbb{E}\left[\left|X_m^{(h)} - X(t_m)\right|\right] = 0$$

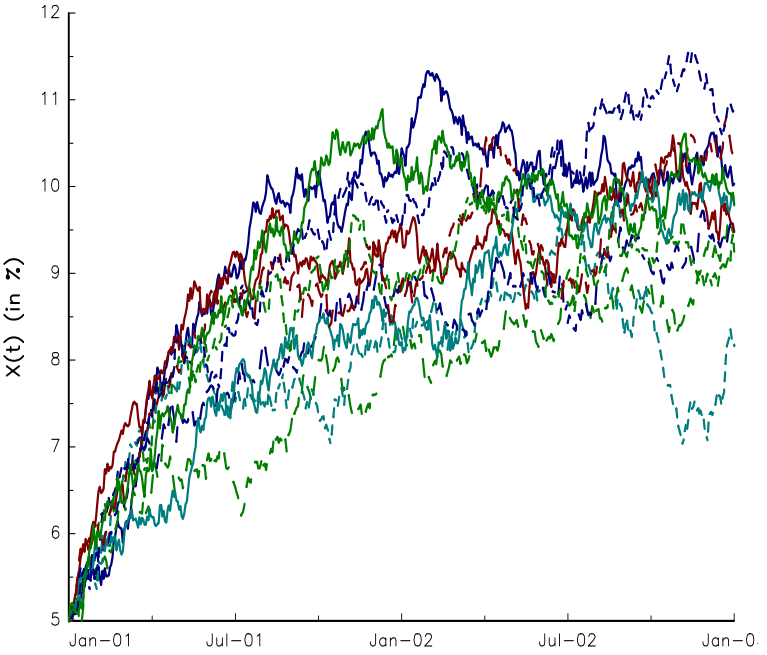

FIGURE 13.20: Simulation of the Ornstein-Uhlenbeck process

for a time t_m. The order of convergence is given by the convergence rate p:

$$\mathbb{E}\left[\left|X_m^{(h)} - X\left(t_m\right)\right|\right] \leq C \cdot h^p$$

where C is a constant and h is sufficiently small ($h \leq h_0$). In the case of the Euler-Maruyama method, the strong order of convergence is 0.5.

Example 147 *For modeling short-term interest rates, Chan et al. (1992) consider the following SDE:*

$$\mathrm{d}X\left(t\right) = \left(\alpha + \beta X\left(t\right)\right)\mathrm{d}t + \sigma X\left(t\right)^{\gamma}\mathrm{d}W\left(t\right) \tag{13.2}$$

We deduce that the fixed-interval Euler-Maruyama scheme is:

$$X_{m+1} = X_m + \left(\alpha + \beta X_m\right)h + \sigma X_m^{\gamma}\sqrt{h}\cdot\varepsilon_m$$

Kloeden and Platen (1992) provided many other approximation schemes, based on Itô-Taylor expansions of the SDE. For instance, the fixed-interval Milstein scheme is:

$$\begin{aligned} X_{m+1} &= X_m + \mu\left(t_m, X_m\right)h + \sigma\left(t_m, X_m\right)\sqrt{h}\cdot\varepsilon_m + \\ &\quad \frac{1}{2}\sigma\left(t_m, X_m\right)\partial_x\sigma\left(t_m, X_m\right)h\left(\varepsilon_m^2 - 1\right) \end{aligned} \tag{13.3}$$

The strong order for the Milstein method is equal to 1, which is better than the Euler-Maruyama method. In terms of implementation, these two approximation schemes remain simple, compared to other Taylor schemes that converge more quickly, but generally use correlated random variables and high order derivatives of the functions $\mu\left(t,x\right)$ and $\sigma\left(t,x\right)$. This is why Euler-Maruyama and Milstein schemes are the most frequent methods used in practice[14].

[14] For instance, one of the most famous methods is the strong order 1.5 Taylor scheme proposed by Platen and Wagner (1982). It requires the second derivatives $\partial_x^2\mu\left(t,x\right)$ and $\partial_x^2\sigma\left(t,x\right)$, and an additional random

If we consider the geometric Brownian motion, the Euler-Maruyama scheme is:

$$X_{m+1} = X_m + \mu X_m h + \sigma X_m \sqrt{h} \cdot \varepsilon_m$$

whereas the Milstein scheme is:

$$\begin{aligned} X_{m+1} &= X_m + \mu X_m h + \sigma X_m \sqrt{h} \cdot \varepsilon_m + \frac{1}{2}\sigma^2 X_m h \left(\varepsilon_m^2 - 1\right) \\ &= X_m + \left(\mu - \frac{1}{2}\sigma^2\right) X_m h + \sigma X_m \sqrt{h} \left(1 + \frac{1}{2}\sigma\sqrt{h}\varepsilon_m\right) \varepsilon_m \end{aligned}$$

It follows that the Milstein scheme operates two corrections for simulating the GBM process:

- the first correction concerns the drift, which is now correct;
- the second correction applies to the diffusion term, which increases if it is positive and decreases if it is negative.

In order to illustrate the differences between these two schemes, we compare them using the same random numbers. A simulation is provided in Figure 13.21 in the case where $\mu = 10\%$ and $\sigma = 50\%$. With a monthly discretization, we notice that the Milstein scheme produces a better solution than the Euler-Maruyama scheme.

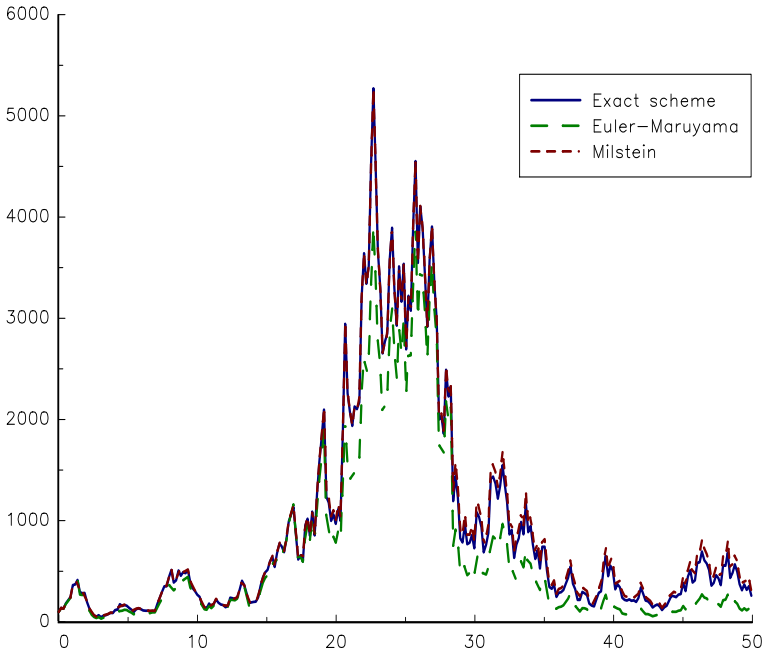

FIGURE 13.21: Comparison of exact, Euler-Maruyama and Milstein schemes (monthly discretization)

When we don't know the analytical solution of $X(t)$, it is natural to simulate the numerical solution of $X(t)$ using Euler-Maruyama and Milstein schemes. However, it may

variable correlated with the increments of the Brownian motion. Even if this scheme is interesting to study from a theoretical point of view, it is never used by practitioners because it is time-consuming.

be sometimes more efficient to find the numerical solution of $Y(t) = f(t, X(t))$ instead of $X(t)$ itself, in particular when $Y(t)$ is more regular than $X(t)$. By Itô's lemma, we have:

$$\begin{aligned} \mathrm{d}Y(t) &= \left(\partial_t f(t,X) + \mu(t,X)\,\partial_x f(t,X) + \frac{1}{2}\sigma^2(t,X)\,\partial_x^2 f(t,X)\right)\mathrm{d}t + \\ &\quad \sigma(t,X)\,\partial_x f(t,X)\,\mathrm{d}W(t) \end{aligned}$$

By using the inverse function $X(t) = f^{-1}(t, Y(t))$, we obtain:

$$\mathrm{d}Y(t) = \mu'(t,Y)\,\mathrm{d}t + \sigma'(t,Y)\,\mathrm{d}W(t)$$

where $\mu'(t,Y)$ and $\sigma'(t,Y)$ are functions of $\mu(t,X)$, $\sigma(t,X)$ and $f(t,X)$. We can then simulate the solution of $Y(t)$ using an approximation scheme and deduce the numerical solution of $X(t)$ by applying the transformation method:

$$X_m = f^{-1}(t_m, Y_m)$$

Let us consider the geometric Brownian motion $X(t)$. The solution of $Y(t) = \ln X(t)$ is equal to:

$$\mathrm{d}Y(t) = \left(\mu - \frac{1}{2}\sigma^2\right)\mathrm{d}t + \sigma\,\mathrm{d}W(t)$$

We deduce that the Euler-Maruyama (or Milstein[15]) scheme with fixed-interval times is:

$$Y_{m+1} = Y_m + \left(\mu - \frac{1}{2}\sigma^2\right)h + \sigma\sqrt{h}\cdot\varepsilon_m$$

It follows that:

$$\ln X_{m+1} = \ln X_m + \left(\mu - \frac{1}{2}\sigma^2\right)h + \sigma\sqrt{h}\cdot\varepsilon_m \tag{13.4}$$

We conclude that this numerical solution is the exact solution (13.1) of the geometric Brownian motion.

The previous application is not interesting, because we know the analytical solution. The approach is more adapted for stochastic differential equations without explicit solutions, for example the Cox-Ingersoll-Ross process:

$$\mathrm{d}X(t) = (\alpha + \beta X(t))\,\mathrm{d}t + \sigma\sqrt{X(t)}\,\mathrm{d}W(t) \tag{13.5}$$

This process is a special case of the CKLS process (13.2) with $\gamma = 1/2$ and can be viewed as an Ornstein-Uhlenbeck process[16] with a reflection at $X(t) = 0$. Using the transformation $Y(t) = \sqrt{X(t)}$, we obtain the following SDE[17]:

$$\begin{aligned} \mathrm{d}Y(t) &= \left(\frac{1}{2}\frac{(\alpha + \beta X(t))}{\sqrt{X(t)}} - \frac{1}{8}\frac{\sigma^2 X(t)}{X(t)^{3/2}}\right)\mathrm{d}t + \frac{1}{2}\frac{\sigma\sqrt{X(t)}}{\sqrt{X(t)}}\,\mathrm{d}W(t) \\ &= \frac{1}{2Y(t)}\left(\alpha + \beta Y^2(t) - \frac{1}{4}\sigma^2\right)\mathrm{d}t + \frac{1}{2}\sigma\,\mathrm{d}W(t) \end{aligned}$$

[15] Because $\partial_y \sigma'(t,Y) = 0$.

[16] The drift can be written as $\alpha + \beta X(t) = -\beta\left(-\alpha\beta^{-1} - X(t)\right)$. We deduce that $a = -\beta$ and $b = -\alpha\beta^{-1}$.

[17] We have $\partial_x f(t,x) = \frac{1}{2}x^{-1/2}$ and $\partial_x^2 f(t,x) = -\frac{1}{4}x^{-3/2}$.

We deduce that the Euler-Maruyama scheme of $Y(t)$ is:

$$Y_{m+1} = Y_m + \frac{1}{2Y_m}\left(\alpha + \beta Y_m^2 - \frac{1}{4}\sigma^2\right)h + \frac{1}{2}\sigma\sqrt{h}\cdot\varepsilon_m$$

It follows that:

$$X_{m+1} = \left(\sqrt{X_m} + \frac{1}{2\sqrt{X_m}}\left(\alpha + \beta X_m - \frac{1}{4}\sigma^2\right)h + \frac{1}{2}\sigma\sqrt{h}\cdot\varepsilon_m\right)^2$$

We can show that this approximation is better than the Euler-Maruyama or Milstein approximation directly applied to the SDE (13.5).

Remark 158 *Generally, we choose the Lamperti transform $Y(t) = f(X(t))$ in order to obtain a constant diffusion term ($\partial_y \sigma'(t,y) = 0$). This implies that:*

$$f(x) = c\int_a^x \frac{1}{\sigma(t,u)}\,\mathrm{d}u$$

Because we have $\partial_x f(t,x) = c/\sigma(t,x)$ and $\partial_x^2 f(t,x) = -c\partial_x\sigma(t,x)/\sigma^2(t,x)$, we obtain:

$$\mathrm{d}Y(t) = c\left(\frac{\mu(t,X)}{\sigma(t,X)} - \frac{\partial_x\sigma(t,X)}{2}\right)\mathrm{d}t + c\,\mathrm{d}W(t)$$

In this case, the Euler-Maruyama scheme coincides with the Milstein scheme. Most of the time, the approximation $X_m = f^{-1}(Y_m)$ gives better results than those obtained with a Milstein method applied to the process $X(t)$.

13.2.2.5 Poisson processes

We have seen that simulating a Poisson process $N(t)$ with constant intensity λ is straightforward, because the inter-arrival times are independent and exponentially distributed with parameter λ. Let t_m be the time when the m^{th} event occurs. The numerical algorithm is then:

1. we set $t_0 = 0$ and $N(t_0) = 0$;
2. we generate a uniform random variate u and calculate the random variate $e \sim \mathcal{E}(\lambda)$ with the formula:
$$e = -\frac{\ln u}{\lambda}$$
3. we update the Poisson process with:
$$t_{m+1} \leftarrow t_m + e \qquad \text{and} \qquad N(t_{m+1}) \leftarrow N(t_m) + 1$$
4. we go back to step 2.

We can also use this algorithm to simulate mixed Poisson process (MPP), which are defined as Poisson process with stochastic intensity Λ. In this case, the algorithm is initialized with a realization λ of the random intensity Λ. On the contrary, this method is not valid in the case of non-homogenous Poisson process (NHPP), where the intensity $\lambda(t)$ varies with time[18]. However, we can show that the inter-arrival times remain independent and exponentially distributed with:

$$\Pr\{T_1 > t\} = \exp(-\Lambda(t))$$

[18]Indeed, we don't know the value of the intensity to use at the second step of the algorithm.

where T_1 is the duration of the first event and $\Lambda(t)$ is the integrated intensity function:

$$\Lambda(t) = \int_0^t \lambda(s)\, \mathrm{d}s$$

It follows that:

$$\Pr\left\{T_1 > \Lambda^{-1}(t)\right\} = \exp(-t) \Leftrightarrow \Pr\left\{\Lambda(T_1) > t\right\} = \exp(-t)$$

We deduce that if $\{t_1, t_2, \ldots, t_M\}$ are the occurrence times of the NHPP of intensity $\lambda(t)$, then $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_M)\}$ are the occurrence times of the homogeneous Poisson process (HPP) of intensity one. Therefore, the algorithm is:

1. we simulate t'_m the time arrivals of the homogeneous Poisson process with intensity $\lambda = 1$;
2. we apply the transform $t_m = \Lambda^{-1}(t'_m)$.

To implement this algorithm, we need to compute the inverse function $\Lambda^{-1}(t)$. When there is no analytical expression, this algorithm may be time-consuming, in particular when $\Lambda(t)$ is calculated with a method of numerical integration. Another approach consists in using the acceptance-rejection algorithm for simulating the NHPP over the period $[0, T]$:

1. we set $\lambda^+ = \max_{t \leq T} \lambda(t)$, $t = 0$, $t_0 = 0$ and $N(t_0) = 0$;
2. we generate a uniform random variate u and calculate the random variate $e \sim \mathcal{E}(\lambda^+)$ with the formula:
$$e = -\frac{\ln u}{\lambda^+}$$
3. we calculate $t = t + e$;
4. if $t > T$, we stop the algorithm;
5. we generate a uniform random variable v; if $v \leq \lambda(t)/\lambda^+$, then we accept the arrival time:
$$t_{m+1} \leftarrow t \qquad \text{and} \qquad N(t_{m+1}) \leftarrow N(t_m) + 1$$
else we reject it;
6. we go back to step 2.

In Figure 13.22, we simulate a non-homogenous Poisson process with the following intensity function:

$$\lambda(t) = 90 + 80 \cdot \sin\left(\frac{6\pi}{5} \cdot t\right)$$

Since $\lambda(t)$ is a cyclical function and $\lambda(t) \in [10, 170]$, the intensity function can vary very quickly. In the bottom/right panel, we draw the histogram of arrival process for the interval $[t, t + \mathrm{d}t]$ and compare it with the expected arrival frequency, which is equal to:

$$\mathbb{E}\left[N(t + \mathrm{d}t) - N(t)\right] = \int_t^{t+\mathrm{d}t} \lambda(s)\, \mathrm{d}s \approx \lambda(t)\, \mathrm{d}t$$

The compound Poisson process $Y(t)$ is defined by:

$$Y(t) = \sum_{i=1}^{N(t)} X_i$$

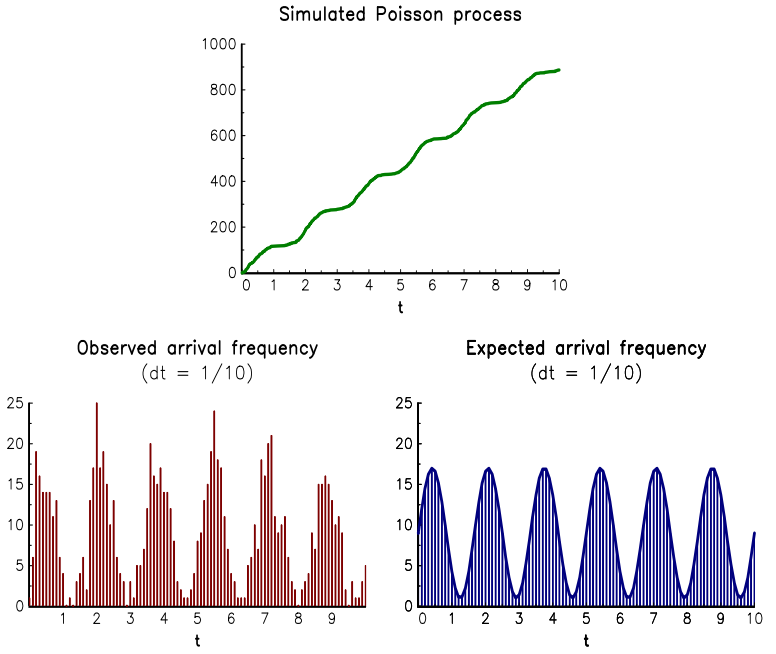

FIGURE 13.22: Simulation of a non-homogenous Poisson process with cyclical intensity

where $N(t)$ is a Poisson process of intensity λ and $\{X_i\}_{i\geq 1}$ is a sequence of *iid* random variables with distribution function $\mathbf{F}$. This process is a generalization of the Poisson process by assuming that jump sizes are not equal to one, but are random. Let $\{t_1, t_2, \ldots, t_M\}$ be the arrival times of the Poisson process. We have:

$$Y(t) = Y(t_m) \qquad \text{if } t \in [t_m, t_{m+1}[$$

and:

$$Y(t_{m+1}) = Y(t_m) + X_{m+1}$$

where X_{m+1} is generated from the distribution function $\mathbf{F}$. Another method to simulate $Y(t)$ is to use the following property: conditionally to $N(T) = n$, the arrival times $\{t_1, t_2, \ldots, t_n\}$ of the Poisson process on the interval $[0, T]$ are distributed as n independent ordered uniform random variables. We deduce this algorithm:

1. we simulate the number n of jumps on the time interval $[0, T]$ by generating a Poisson random variable with parameter λT;
2. we simulate n uniform random variates $(u_1, \ldots, u_n)$ and sort them[19]:

$$u_{1:n} \leq u_{2:n} \leq \ldots \leq u_{n:n}$$

3. we simulate n random variates $(x_1, \ldots, x_n)$ from the probability distribution $\mathbf{F}$;
4. we finally generate the compound Poisson process by:

$$Y(t) = \sum_{i=1}^{n} \mathbb{1}\left\{u_{i:n} \leq \frac{t}{T}\right\} \cdot x_i$$

[19]The arrival times are given by the formula: $t_m = T \cdot u_{m:n}$.

13.2.2.6 Jump-diffusion processes

A jump-diffusion model is a process, which is generally defined by:

$$\mathrm{d}X(t) = \mu(t, X)\,\mathrm{d}t + \sigma(t, X)\,\mathrm{d}W(t) + \eta\left(t^-, X^-\right)\mathrm{d}J(t) \tag{13.6}$$

where $J(t)$ is a jump process. Between two jumps, $\mathrm{d}J(t)$ is equal to zero and the process $X(t)$ is continuous and evolves according to the SDE:

$$\mathrm{d}X(t) = \mu(t, X)\,\mathrm{d}t + \sigma(t, X)\,\mathrm{d}W(t)$$

When a jump occurs, the process is discontinuous and we have:

$$X(t) = X\left(t^-\right) + \eta\left(t^-, X\left(t^-\right)\right)\mathrm{d}J(t)$$

The jump process may be a Poisson process $N(t)$ with intensity λ or a compound Poisson process $Y(t) = \sum_{i=1}^{N(t)} Z_i$ where $\{Z_i\}_{i\geq 1}$ is a sequence of *iid* random variables with distribution $\mathbf{F}$.

In the case $J(t) = N(t)$, the Euler scheme is:

$$\begin{aligned} X(t_{m+1}) &= X(t_m) + \mu(t_m, X(t_m)) \cdot (t_{m+1} - t_m) + \\ &\quad \sigma(t_m, X(t_m)) \cdot (W(t_{m+1}) - W(t_m)) + \\ &\quad \eta(t_m, X(t_m)) \cdot (N(t_{m+1}) - N(t_m)) \end{aligned}$$

We finally obtain:

$$\begin{aligned} X_{m+1} &= X_m + \mu(t_m, X_m) \cdot (t_{m+1} - t_m) + \\ &\quad \sigma(t_m, X_m) \cdot \sqrt{t_{m+1} - t_m} \cdot \varepsilon_m + \eta(t_m, X_m) \cdot \xi_m \end{aligned} \tag{13.7}$$

where $\varepsilon_m \sim \mathcal{N}(0,1)$ and $\xi_m \sim \mathcal{P}(\lambda(t_{m+1} - t_m))$. We have:

$$\begin{aligned} \Pr\{\xi_m = 0\} &= e^{-\lambda(t_{m+1}-t_m)} \\ \Pr\{\xi_m = 1\} &= e^{-\lambda(t_{m+1}-t_m)}\lambda(t_{m+1} - t_m) \\ \Pr\{\xi_m \geq 2\} &= 1 - e^{-\lambda(t_{m+1}-t_m)}(1 + \lambda(t_{m+1} - t_m)) \end{aligned}$$

If we assume that the stepsize $t_{m+1} - t_m$ is small, we obtain $\Pr\{\xi_m = 0\} \approx 1 - \lambda(t_{m+1} - t_m)$, $\Pr\{\xi_m = 1\} \approx \lambda(t_{m+1} - t_m)$ and $\Pr\{\xi_m \geq 2\} \approx 0$. Therefore, we can generate ξ_m by:

$$\xi_m = \begin{cases} 1 & \text{if } \lambda(t_{m+1} - t_m) \leq u_m \\ 0 & \text{otherwise} \end{cases}$$

where u_m is a uniform random variate. Another way to simulate $X(t)$ is to first simulate the arrival times of the Poisson process. We denote these times by $\tau_1, \tau_2, \ldots, \tau_N$, and combine this grid with the initial grid $t_1, t_2, \ldots, t_M$. We then apply the Euler scheme (13.7) on the augmented grid, but we are now sure that we cannot have more than one jump between two discretization times. We illustrate this algorithm by considering the SDE:

$$\mathrm{d}X(t) = 0.15 \cdot X(t)\,\mathrm{d}t + 0.20 \cdot X(t)\,\mathrm{d}W(t) + \left(30 - 0.30 \cdot X\left(t^-\right)\right) \cdot \mathrm{d}J(t)$$

A simulated path is given in Figure 13.23, where the jumps are indicated by a dashed line.

In the case of the compound Poisson process $J(t) = Y(t)$, we can obtain explicit solutions for some processes. For instance, the model of Merton (1976) considers that the continuous part is a geometric Brownian motion:

$$\mathrm{d}X(t) = \mu X(t)\,\mathrm{d}t + \sigma X(t)\,\mathrm{d}W(t) + X\left(t^-\right)\mathrm{d}J(t)$$

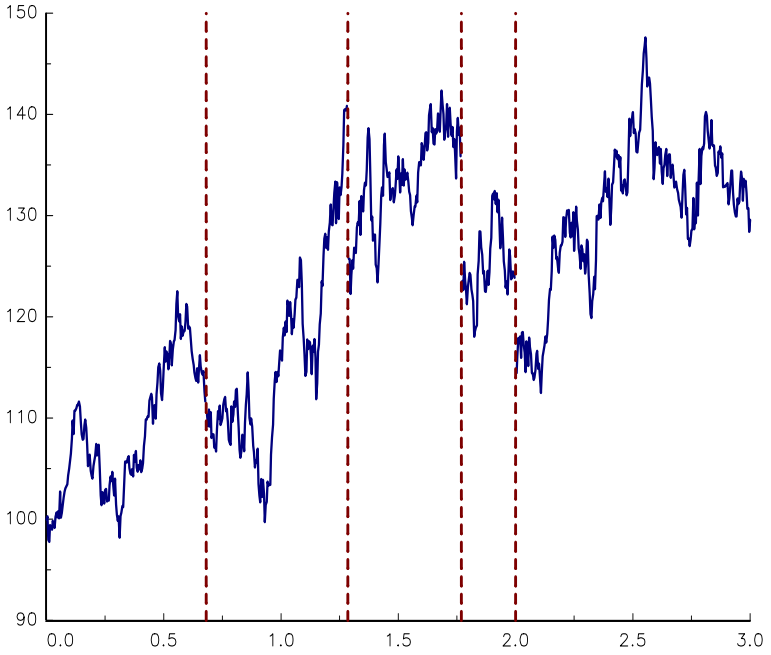

FIGURE 13.23: Simulation of a jump-diffusion process

If we assume that the i^{th} jump occurs at time t, we obtain[20]:

$$\begin{aligned} X(t) &= X(t^-) + X(t^-) Z_i \\ &= (1 + Z_i) X(t^-) \end{aligned}$$

We deduce that:

$$\begin{aligned} X(t) &= X(0) \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W(t) + J(t)\right) \\ &= X(0) e^{\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W(t)} \prod_{i=1}^{N(t)} (1 + Z_i) \end{aligned}$$

In the general case, the Euler scheme is:

$$\begin{aligned} X_{m+1} &= X_m + \mu(t_m, X_m) \cdot (t_{m+1} - t_m) + \\ &\quad \sigma(t_m, X_m) \cdot \sqrt{t_{m+1} - t_m} \cdot \varepsilon_m + \eta(t_m, X_m) \cdot \xi_m \end{aligned}$$

where $\varepsilon_m \sim \mathcal{N}(0,1)$ and $\xi_m = Y(t_{m+1}) - Y(t_m)$. As we have previously presented an algorithm to generate $Y(t)$, there is no difficulty to simulate $X(t)$.

13.2.2.7 Processes related to Brownian motion

We have previously shown how to simulate a stochastic differential equation by assuming the initial position of the random process. In finance, we also need to simulate stochastic processes with other constraints (Brownian bridge, Brownian meander) or statistics of the SDE (minimum, maximum, stopping time).

[20] We assume that $Z_i > -1$.

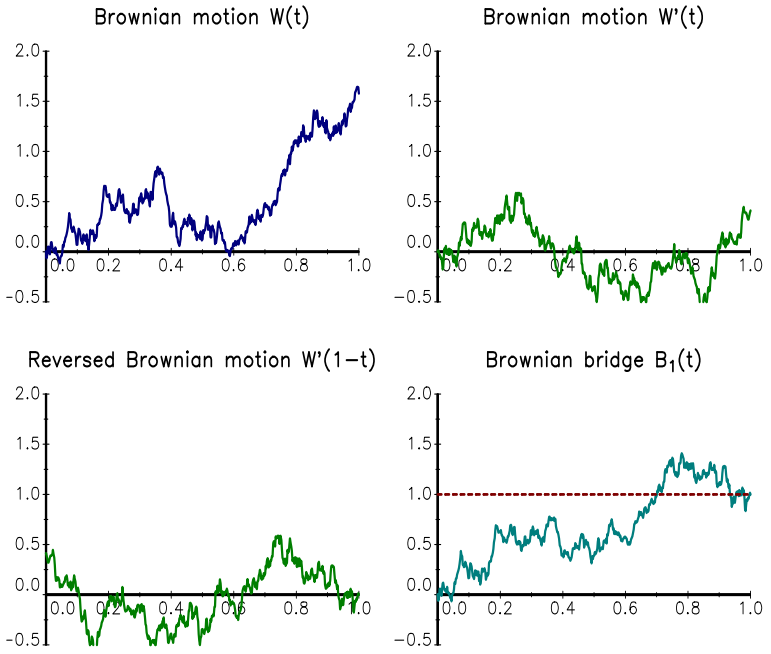

FIGURE 13.24: Simulation of the Brownian bridge $B_1(t)$ using the time reversibility property

A Brownian bridge $B_r(t)$ is a Brownian motion $W(t)$ such that $W(0) = 0$ and $W(1) = r$ (Revuz and Yor, 1999). For $t \in [0, 1]$, we have[21]:

$$B_r(t) = W(t) + (r - W(1)) \cdot t$$

Devroye (2010) noticed that:

$$W(1) = W(t) + (W(1) - W(t))$$

The time reversibility property of the Brownian motion implies that $W(1) - W(t) \overset{\mathcal{L}}{=} W(1-t)$. It follows that:

$$\begin{aligned} B_r(t) &= W(t) + (r - (W(t) + W'(1-t))) \cdot t \\ &= r \cdot t + (1-t) \cdot W(t) + t \cdot W'(1-t) \end{aligned}$$

Figure 13.24 illustrates the simulation of $B_1(t)$ by using two simulated paths $W(t)$ and $W'(t)$. We also notice that:

$$\begin{aligned} B_r(t) &= r \cdot t + (1-t) \cdot \sqrt{t} \cdot \varepsilon_1 + t \cdot \sqrt{1-t} \cdot \varepsilon_2 \\ &= r \cdot t + t \cdot \sqrt{t(1-t)} \cdot \varepsilon \end{aligned}$$

where ε_1, ε_2 and ε are standard Gaussian random variables. If we now assume that $s \leq t \leq u$, $W(s) = w_s$ and $W(u) = w_u$, the Brownian bridge becomes:

$$B(t) = \frac{(u-t) \cdot w_s + (t-s) \cdot w_u}{u-s} + \sqrt{\frac{(t-s) \cdot (u-t)}{u-s}} \cdot \varepsilon$$

[21] We verify that the increments of $B_r(t)$ are independent, $B_r(0) = 0$ and $B_r(1) = r$.

because of the scaling property of the Brownian motion[22]. If we consider the simulation of $B(t)$ for different values $t_m \in [s, u]$, we proceed by filling the path with the iterative algorithm:

1. we initialize the algorithm with $m = 1$;
2. we simulate the Brownian bridge $B(t_m)$ such that $B(s) = w_s$ and $B(u) = w_u$;
3. we set $s = t_m$ and $B(s) = B(t_m)$;
4. we go back to step 2.

In Figure 13.25, we report 5 simulations of the Brownian Bridge $B(t)$ such that $B(0) = 0$, $B(1) = 1$, $B(3) = 3$ and $B(5) = 2$.

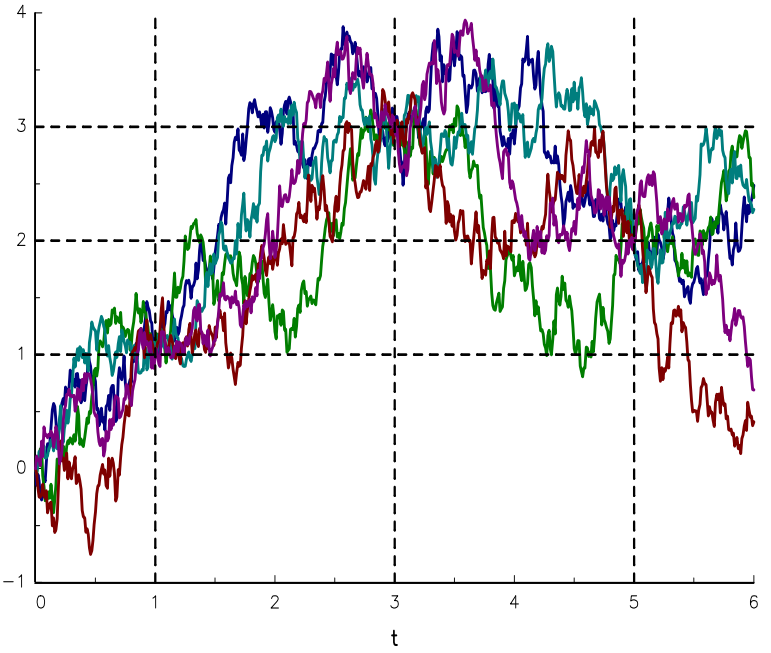

FIGURE 13.25: Simulation of the Brownian bridge $B(t)$

To simulate a process $X(t)$ with fixed values at times $\tau_1, \ldots, \tau_p$, we assume that we have an explicit solution $X(t) = g(W(t))$ implying that $W(t) = g^{-1}(X(t))$. Simulating a diffusion bridge $X(t)$ consists then in generating the Brownian bridge $B(t)$ such that $W(\tau_i) = g^{-1}(X(\tau_i))$, and applying the transformation $X(t) = g(B(t))$. For instance, if we consider the geometric Brownian motion, we have:

$$X(t) = g(W(t)) = x_0 \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W(t)\right)$$

and:

$$W(t) = g^{-1}(X(t)) = \frac{\ln X(t) - \ln x_0 - \left(\mu - \frac{1}{2}\sigma^2\right)t}{\sigma}$$

[22] See also Exercise 13.4.8 on page 891 for an alternative proof (Glasserman, 2003).

We assume that $x_0 = 100$, $\mu = 0$ and $\sigma = 10\%$. The fixed values of $X(t)$ are given in the table below. Using the previous formula, we deduce the values taken by the Brownian bridge:

τ_j	$X(\tau_i)$	$W(\tau_i)$
0	100	0.0000
1	110	1.0031
3	100	0.1500
5	90	−0.8036

We have reported five simulated path of this diffusion bridge in Figure 13.26.

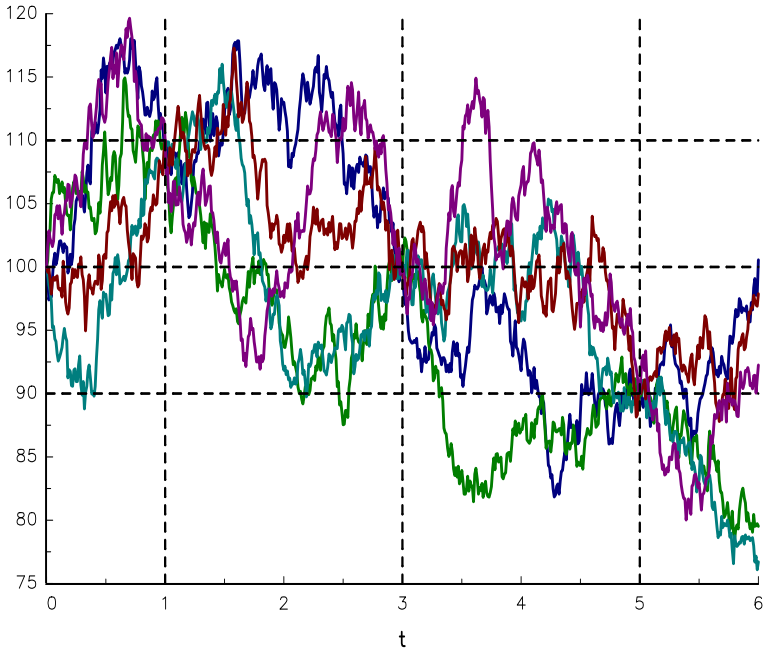

FIGURE 13.26: Simulation of the diffusion bridge $X(t)$

Diffusion bridges are important in finance when we would like to study extremes of a diffusion process. If we want to find the maximum of the stochastic process $X(t)$ over $[0, T]$, we can simulate $X(t)$ and take the maximum of the generated path:

$$\hat{M} = \max_m X_m$$

Another approach consists in locating the maximum:

$$m^\star = \arg\max_m X_m$$

and simulating the diffusion bridge $B(t)$ such that $X(t_{m^\star-1}) = X_{m^\star-1}$, $X(t_{m^\star}) = X_{m^\star}$ and $X(t_{m^\star+1}) = X_{m^\star+1}$. In this case, we can define another estimator of the maximum:

$$\tilde{M} = \max_{t \in \left[t_{m^\star-1}, t_{m^\star+1}\right]} B(t)$$

By construction, we always have $\tilde{M} \geq \hat{M}$. For instance, we report the probability density function of $\hat{M}$ and $\tilde{M}$ in Figure 13.27 when we consider the geometric Brownian motion

with $x_0 = 100$, $\mu = 0$, $\sigma = 15\%$ and $T = 1$. The GBM process has been simulated with a fixed stepsize $h = 0.1$, whereas the diffusion bridge has been simulated with $h = 0.001$. This implies that each path uses $1/0.1 = 10$ discretization points in the first case and $10 + 0.2/0.001 = 210$ discretization points in the second case. The estimation based on the diffusion bridge is then equivalent to consider a scheme with $1/0.001 = 1\,000$ discretization points.

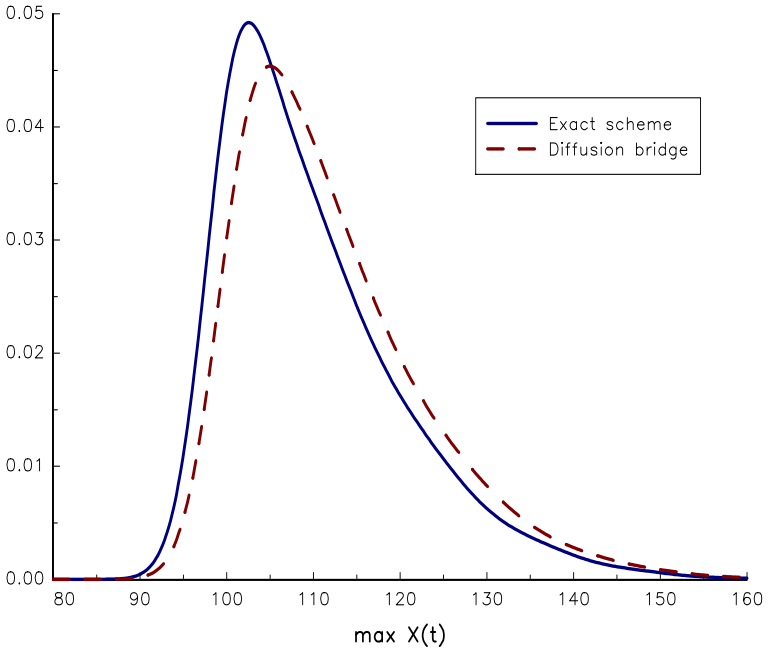

FIGURE 13.27: Density of the maximum estimators $\hat{M}$ and $\tilde{M}$

Remark 159 *In the case of the geometric Brownian motion $X(t)$, the distribution function of the maximum is known. Indeed, we have*[23]*:*

$$\Pr\{M(t) \geq x\} = \exp\left(\frac{2\eta x}{\sigma^2}\right)\Phi\left(\frac{-x-\eta t}{\sigma\sqrt{t}}\right) + \Phi\left(\frac{-x+\eta t}{\sigma\sqrt{t}}\right)$$

where $M(t)$ is the maximum of a Brownian motion with a constant drift:

$$M(t) = \max_{s\leq t} \eta t + \sigma W(t)$$

We notice that:

$$\ln\frac{X(t)}{X(0)} = \left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W(t)$$

[23] In the case of the minimum, we can use the following identity:

$$m(t) = \min_{s\leq t} \eta t + \sigma W(t) = -\max_{s\leq t} -\eta t - \sigma W(t)$$

It follows that:

$$\Pr\{m(t) \leq x\} = \exp\left(\frac{2\eta x}{\sigma^2}\right)\Phi\left(\frac{x+\eta t}{\sigma\sqrt{t}}\right) + \Phi\left(\frac{x-\eta t}{\sigma\sqrt{t}}\right)$$

It follows that:

$$\begin{aligned}\Pr\left\{\max_{s\leq t} X(s) \geq x\right\} &= \Pr\left\{\max_{s\leq t} \ln X(s) \geq \ln x\right\} \\ &= \Pr\left\{M(t) \geq \ln x - \ln x_0\right\}\end{aligned}$$

where $\eta = \mu - \frac{1}{2}\sigma^2$.

Remark 160 *Diffusion bridges are extensively used when pricing look-back options by Monte Carlo, but also barrier options. Indeed, we need to locate precisely the stopping time when the process crosses the barrier. More generally, they may accelerate the convergence of Monte Carlo methods in the case of path-dependent derivatives (Glasserman, 2003).*

13.2.3 Multivariate continuous-time processes

13.2.3.1 Multidimensional Brownian motion

Let $W(t) = (W_1(t), \ldots, W_n(t))$, be a n-dimensional Brownian motion. Each component $W_i(t)$ is a Brownian motion:

$$W_i(t) - W_i(s) \sim \mathcal{N}(0, t-s)$$

Moreover, we have:

$$\mathbb{E}\left[W_i(t) W_j(s)\right] = \min(t,s) \cdot \rho_{i,j}$$

where $\rho_{i,j}$ is the correlation between the two Brownian motions W_i and W_j. We deduce that:

$$\begin{cases} W(0) = \mathbf{0} \\ W(t) = W(s) + \epsilon(s,t) \end{cases}$$

where $\epsilon(s,t) \sim \mathcal{N}_n(\mathbf{0}, (t-s)\,\rho)$ are *iid* random vectors. It follows that the numerical solution is:

$$W_{m+1} = W_m + \sqrt{t_{m+1} - t_m} \cdot P \cdot \varepsilon_m$$

where P is the Cholesky decomposition of the correlation matrix ρ and $\varepsilon_m \sim \mathcal{N}_n(0, I)$ are *iid* random vectors. In the case of fixed-interval times, the recursion becomes:

$$W_{m+1} = W_m + \sqrt{h} \cdot P \cdot \varepsilon_m$$

In Figures 13.28 and 13.29, we simulate the realization of two-dimensional Brownian motions. Since the two simulated paths use the same random numbers, the difference comes from the correlation $\rho_{1,2}$, which is equal to zero for the first case and 85% for the second case.

13.2.3.2 Multidimensional geometric Brownian motion

Let us now consider the multidimensional geometric Brownian motion[24]:

$$\begin{cases} \mathrm{d}X(t) = \mu \odot X(t)\ \mathrm{d}t + \mathrm{diag}\left(\sigma \odot X(t)\right) \mathrm{d}W(t) \\ X(0) = x_0 \end{cases}$$

[24]The symbol $\odot$ is the Hadamard product.

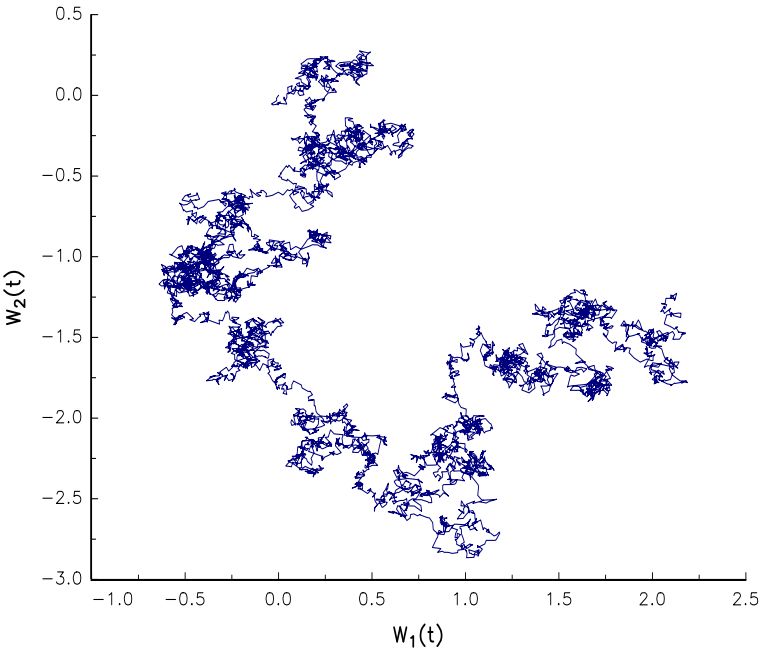

FIGURE 13.28: Brownian motion in the plane (independent case)

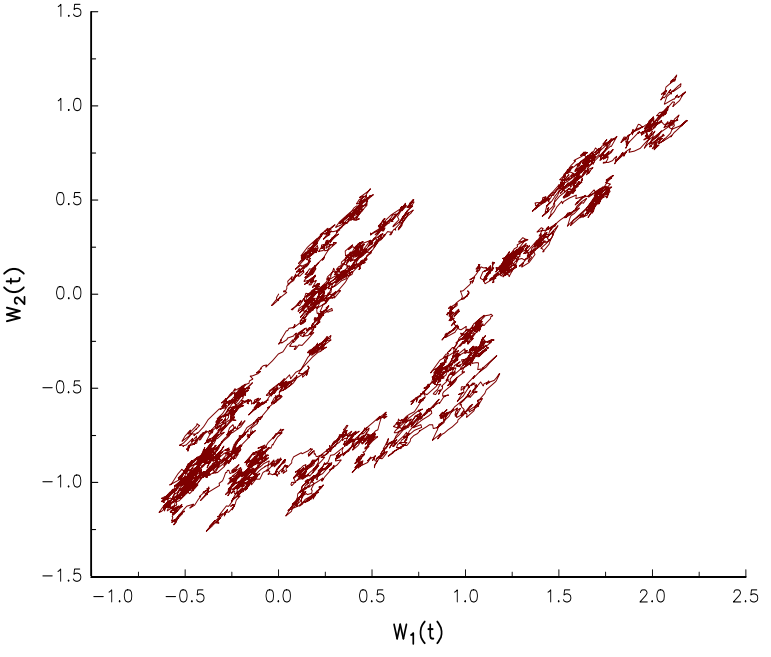

FIGURE 13.29: Brownian motion in the plane ($\rho_{1,2} = 85\%$)

where $X(t) = (X_1(t), \ldots, X_n(t))$, $\mu = (\mu_1, \ldots, \mu_n)$, $\sigma = (\sigma_1, \ldots, \sigma_n)$ and $W(t) = (W_1(t), \ldots, W_n(t))$ is a n-dimensional Brownian motion with $\mathbb{E}\left[W(t) W(t)^\top\right] = \rho t$. If we consider the j^{th} component of $X(t)$, we have:

$$\mathrm{d}X_j(t) = \mu_j X_j(t)\,\mathrm{d}t + \sigma_j X_j(t)\,\mathrm{d}W_j(t)$$

The solution of the multidimensional SDE is a multivariate log-normal process with:

$$X_j(t) = X_j(0) \cdot \exp\left(\left(\mu_j - \frac{1}{2}\sigma_j^2\right) t + \sigma_j W_j(t)\right)$$

where $W(t) \sim \mathcal{N}_n(0, \rho t)$. We deduce that the exact scheme to simulate the multivariate GBM is:

$$\begin{cases} X_{1,m+1} = X_{1,m} \cdot \exp\left(\left(\mu_1 - \frac{1}{2}\sigma_1^2\right)(t_{m+1} - t_m) + \sigma_1\sqrt{t_{m+1} - t_m} \cdot \varepsilon_{1,m}\right) \\ \vdots \\ X_{j,m+1} = X_{j,m} \cdot \exp\left(\left(\mu_j - \frac{1}{2}\sigma_j^2\right)(t_{m+1} - t_m) + \sigma_j\sqrt{t_{m+1} - t_m} \cdot \varepsilon_{j,m}\right) \\ \vdots \\ X_{n,m+1} = X_{n,m} \cdot \exp\left(\left(\mu_n - \frac{1}{2}\sigma_n^2\right)(t_{m+1} - t_m) + \sigma_n\sqrt{t_{m+1} - t_m} \cdot \varepsilon_{n,m}\right) \end{cases}$$

where $(\varepsilon_{1,m}, \ldots, \varepsilon_{n,m}) \sim \mathcal{N}_n(\mathbf{0}, \rho)$.

Remark 161 *Monte Carlo methods extensively use this scheme for calculating the price of multi-asset derivatives in the Black-Scholes model.*

13.2.3.3 Euler-Maruyama and Milstein schemes

We consider the general SDE:

$$\begin{cases} \mathrm{d}X(t) = \mu(t, X(t))\,\mathrm{d}t + \sigma(t, X(t))\,\mathrm{d}W(t) \\ X(0) = x_0 \end{cases}$$

where $X(t)$ and $\mu(t, X(t))$ are $n \times 1$ vectors, $\sigma(t, X(t))$ is a $n \times p$ matrix and $W(t)$ is a $p \times 1$ vector. We assume that $\mathbb{E}\left[W(t) W(t)^\top\right] = \rho t$, where ρ is a $p \times p$ correlation matrix. The corresponding Euler-Maruyama scheme is:

$$X_{m+1} = X_m + \mu(t_m, X_m) \cdot (t_{m+1} - t_m) + \sigma(t_m, X_m)\sqrt{t_{m+1} - t_m} \cdot \varepsilon_m$$

where $\varepsilon_m \sim \mathcal{N}_p(0, \rho)$. In the case of a diagonal system[25], we retrieve the one-dimensional scheme:

$$X_{j,m+1} = X_{j,m} + \mu_j(t_m, X_{j,m}) \cdot (t_{m+1} - t_m) + \sigma_{j,j}(t_m, X_{j,m}) \cdot \sqrt{t_{m+1} - t_m}\varepsilon_{j,m}$$

However, the random variables $\varepsilon_{j,m}$ and $\varepsilon_{j',m}$ may be correlated.

Example 148 *We consider the Heston model:*

$$\begin{cases} \mathrm{d}X(t) = \mu X(t)\,\mathrm{d}t + \sqrt{v(t)}X(t)\,\mathrm{d}W_1(t) \\ \mathrm{d}v(t) = a(b - v(t))\,\mathrm{d}t + \sigma\sqrt{v(t)}\,\mathrm{d}W_2(t) \end{cases}$$

[25] This means that $\mu_j(t, x) = \mu_j(t, x_j)$ and $\sigma(t, x)$ is a $n \times n$ diagonal matrix with $\sigma_{j,j}(t, x) = \sigma_{j,j}(t, x_j)$.

where $\mathbb{E}[W_1(t) W_2(t)] = \rho t$. *By applying the fixed-interval Euler-Maruyama scheme to* $(\ln X(t), v(t))$, *we obtain:*

$$\ln X_{m+1} = \ln X_m + \left(\mu - \frac{1}{2} v_m\right) h + \sqrt{v_m h} \cdot \varepsilon_{1,m}$$

and[26]*:*

$$v_{m+1} = v_m + a(b - v_m) h + \sigma \sqrt{v_m h} \cdot \varepsilon_{2,m}$$

Here, $\varepsilon_{1,m}$ *and* $\varepsilon_{2,m}$ *are two standard Gaussian random variables with correlation* ρ.

The multidimensional version of the Milstein scheme is[27]:

$$\begin{aligned} X_{j,m+1} &= X_{j,m} + \mu_j(t_m, X_m)(t_{m+1} - t_m) + \sum_{k=1}^{p} \sigma_{j,k}(t_m, X_m) \Delta W_{k,m} + \\ &\quad \sum_{k=1}^{p} \sum_{k'=1}^{p} \mathcal{L}^{(k)} \sigma_{j,k'}(t_m, X_m) \mathcal{I}_{(k,k')} \end{aligned}$$

where:

$$\mathcal{L}^{(k)} f(t,x) = \sum_{k''=1}^{n} \sigma_{k'',k}(t_m, X_m) \frac{\partial f(t,x)}{\partial x_{k''}}$$

and:

$$\mathcal{I}_{(k,k')} = \int_{t_m}^{t_{m+1}} \int_{t_m}^{s} \mathrm{d}W_k(t)\, \mathrm{d}W_{k'}(s)$$

In the case of a diagonal system, the Milstein scheme may be simplified as follows:

$$\begin{aligned} X_{j,m+1} &= X_{j,m} + \mu_j(t_m, X_{j,m})(t_{m+1} - t_m) + \sigma_{j,j}(t_m, X_{j,m}) \Delta W_{j,m} + \\ &\quad \mathcal{L}^{(j)} \sigma_{j,j}(t_m, X_{j,m}) \mathcal{I}_{(j,j)} \end{aligned}$$

where[28]:

$$\begin{aligned} \mathcal{I}_{(j,j)} &= \int_{t_m}^{t_{m+1}} \int_{t_m}^{s} \mathrm{d}W_j(t)\, \mathrm{d}W_j(s) \\ &= \int_{t_m}^{t_{m+1}} (W_j(s) - W_j(t_m))\, \mathrm{d}W_j(s) \\ &= \frac{1}{2}\left((\Delta W_{j,m})^2 - (t_{m+1} - t_m)\right) \end{aligned}$$

We deduce that the Milstein scheme is:

$$\begin{aligned} X_{j,m+1} &= X_{j,m} + \mu_j(t_m, X_{j,m})(t_{m+1} - t_m) + \\ &\quad \sigma_{j,j}(t_m, X_{j,m}) \sqrt{t_{m+1} - t_m} \varepsilon_{j,m} + \\ &\quad \frac{1}{2} \sigma_{j,j}(t_m, X_{j,m}) \partial_{x_j} \sigma_{j,j}(t_m, X_{j,m})(t_{m+1} - t_m)\left(\varepsilon_{j,m}^2 - 1\right) \end{aligned}$$

[26] To avoid that v_{m+1} is negative, we can use the truncation method:

$$v_{m+1} \leftarrow \max(v_{m+1}, 0)$$

or the reflection method:

$$v_{m+1} \leftarrow |v_{m+1}|$$

[27] We have $\Delta W_{k,m} = W_k(t_{m+1}) - W_k(t_m)$.

[28] By applying Itô's lemma to $Y_t = \frac{1}{2}\left(W_j(t)^2 - t\right)$, we obtain $\mathrm{d}Y(t) = W_j(t)\, \mathrm{d}W_j(t)$.

We obtain the same expression as the formula given by Equation (13.3), except that the random variables $\varepsilon_{j,m}$ and $\varepsilon_{j',m}$ may now be correlated.

Example 149 *If we apply the fixed-interval Milstein scheme to the Heston model, we obtain:*

$$\ln X_{m+1} = \ln X_m + \left(\mu - \frac{1}{2}v_m\right) h + \sqrt{v_m h} \cdot \varepsilon_{1,m}$$

and:

$$v_{m+1} = v_m + a\left(b - v_m\right) h + \sigma\sqrt{v_m h} \cdot \varepsilon_{2,m} + \frac{1}{4}\sigma^2 h\left(\varepsilon_{2,m}^2 - 1\right)$$

Here, $\varepsilon_{1,m}$ and $\varepsilon_{2,m}$ are two standard Gaussian random variables with correlation ρ.

Remark 162 *The multidimensional Milstein scheme is generally not used, because the terms $\mathcal{L}^{(k)}\sigma_{j,k'}\left(t_m, X_m\right)\mathcal{I}_{(k,k')}$ are complicated to simulate. For the Heston model, we obtain a very simple scheme, because we only apply the Milstein scheme to the process $v\left(t\right)$ and not to the vector process $\left(\ln X\left(t\right), v\left(t\right)\right)$. If we also apply the Milstein scheme to $\ln X\left(t\right)$, we obtain:*

$$\ln X_{m+1} = \ln X_m + \left(\mu - \frac{1}{2}v_m\right) h + \sqrt{v_m h} \cdot \varepsilon_{1,m} + A_m$$

where:

$$\begin{aligned} A_m &= \sum_{k=1}^{2}\sum_{k'=1}^{2}\left(\sum_{k''=1}^{2}\sigma_{k'',k}\left(t_m, X_m\right)\frac{\sigma_{1,k'}\left(t_m, X_m\right)}{\partial\, x_{k''}}\right)\mathcal{I}_{(k,k')} \\ &= \sigma\sqrt{v\left(t\right)} \cdot \frac{1}{2\sqrt{v\left(t\right)}} \cdot \mathcal{I}_{(2,1)} \\ &= \frac{\sigma}{2} \cdot \mathcal{I}_{(2,1)} \end{aligned}$$

Let $W_2\left(t\right) = \rho W_1\left(t\right) + \sqrt{1-\rho^2}W^\star\left(t\right)$ where $W^\star\left(t\right)$ is a Brownian motion independent from $W_1\left(t\right)$. It follows that:

$$\begin{aligned} \mathcal{I}_{(2,1)} &= \int_{t_m}^{t_{m+1}}\int_{t_m}^{s} \mathrm{d}W_2\left(t\right)\mathrm{d}W_1\left(s\right) \\ &= \int_{t_m}^{t_{m+1}}\left(\rho W_1\left(s\right) + \sqrt{1-\rho^2}W^\star\left(s\right)\right)\mathrm{d}W_1\left(s\right) - \\ &\quad \int_{t_m}^{t_{m+1}}\left(\rho W_1\left(t_m\right) + \sqrt{1-\rho^2}W^\star\left(t_m\right)\right)\mathrm{d}W_1\left(s\right) \\ &= \rho\int_{t_m}^{t_{m+1}}\left(W_1\left(s\right) - W_1\left(t_m\right)\right)\mathrm{d}W_1\left(s\right) + \\ &\quad \sqrt{1-\rho^2}\int_{t_m}^{t_{m+1}}\left(W^\star\left(s\right) - W^\star\left(t_m\right)\right)\mathrm{d}W_1\left(s\right) \end{aligned}$$

and:

$$\mathcal{I}_{(2,1)} = \frac{1}{2}\rho\left(\left(\Delta W_{1,m}\right)^2 - \left(t_{m+1} - t_m\right)\right) + B_m$$

We finally deduce that the multidimensional Milstein scheme of the Heston model is:

$$\ln X_{m+1} = \ln X_m + \left(\mu - \frac{1}{2}v_m\right) h + \sqrt{v_m h} \cdot \varepsilon_{1,m} + \frac{1}{4}\rho\sigma h\left(\varepsilon_{1,m}^2 - 1\right) + B_m$$

and:

$$v_{m+1} = v_m + a\left(b - v_m\right)h + \sigma\sqrt{v_m h}\cdot\varepsilon_{2,m} + \frac{1}{4}\sigma^2 h\left(\varepsilon_{2,m}^2 - 1\right)$$

where B_m is a correction term defined by:

$$B_m = \sqrt{1-\rho^2}\int_{t_m}^{t_{m+1}}\left(W^\star(s) - W^\star(t_m)\right)\mathrm{d}W_1(s)$$

We notice that B_m cannot be explicitly calculate and requires numerical integration to be simulated[29] .

13.3 Monte Carlo methods

At the beginning, the Monte Carlo method is a numerical tool for computing integrals based on the simulation of random variables (Metropolis and Ulam, 1949). By extension, it now defines all numerical methods, which use simulations.

13.3.1 Computing integrals

13.3.1.1 A basic example

One of the early uses of the Monte Carlo method was the numerical calculation of the number π by Bouffon and Laplace. Suppose we have a circle with radius r and a $2r \times 2r$ square of the same center. Since the area of the circle is equal to πr^2, the numerical calculation of π is equivalent to compute the area of the circle with $r = 1$. In this case, the area of the square is 4, and we have[30]:

$$\pi = 4\frac{\mathcal{A}(\text{circle})}{\mathcal{A}(\text{square})}$$

To determine π, we simulate n_S random vectors (u_s, v_s) of uniform random variables $\mathcal{U}_{[-1,1]}$ and we obtain:

$$\pi = \lim_{n_S\to\infty} 4\frac{n_c}{n}$$

where n_c is the number of points (u_s, v_s) in the circle:

$$n_c = \sum_{s=1}^{n_S}\mathbb{1}\left\{u_s^2 + v_s^2 \leq r^2\right\}$$

We illustrate this numerical computation in Figure 13.30 with 1 000 simulated points (u_s, v_s). We indicate by a red cross symbol (resp. by a blue square symbol) the points which are inside (resp. outside) the circle. In this experiment, we obtain $n_c = 802$ and $\pi \simeq 4 \times 802/1\,000 = 3.2080$.

[29] However, B_m is not independent from $\varepsilon_{1,m}$.

[30] In fact, this relationship holds for all values of r.

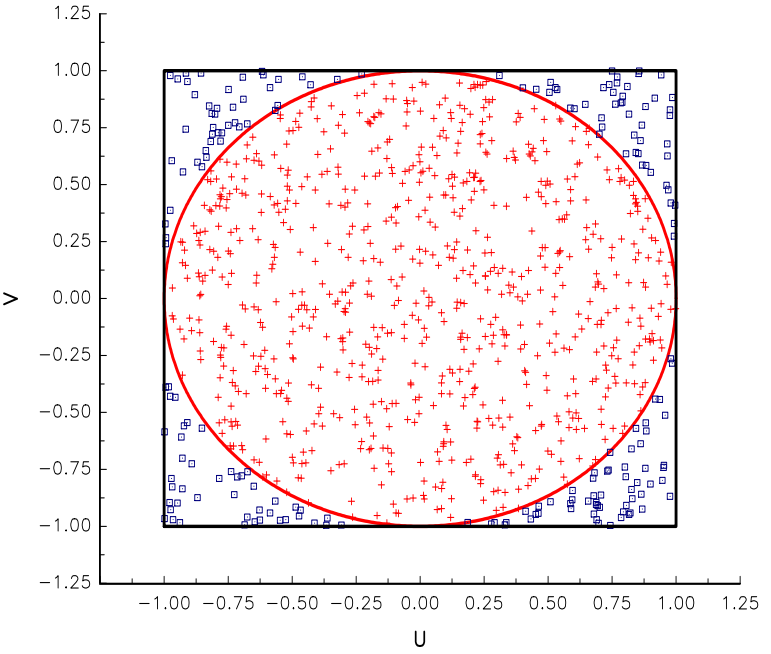

FIGURE 13.30: Computing π with 1 000 simulations

13.3.1.2 Theoretical framework

We consider the multiple integral:

$$I = \int \cdots \int_{\Omega} \varphi(x_1, \ldots, x_n)\, \mathrm{d}x_1 \cdots \mathrm{d}x_n$$

Let $X = (X_1, \ldots, X_n)$ be a uniform random vector with probability distribution $\mathcal{U}_{[\Omega]}$, such that Ω is inscribed within the hypercube $[\Omega]$. By construction, the probability density function is:

$$f(x_1, \ldots, x_n) = 1$$

We deduce that:

$$\begin{aligned} I &= \int \cdots \int_{[\Omega]} \mathbb{1}\{(x_1, \ldots, x_n) \in \Omega\} \cdot \varphi(x_1, \ldots, x_n)\, \mathrm{d}x_1 \cdots \mathrm{d}x_n \\ &= \mathbb{E}\left[\mathbb{1}\{(X_1, \ldots, X_n) \in \Omega\} \cdot \varphi(X_1, \ldots, X_n)\right] \\ &= \mathbb{E}\left[h(X_1, \ldots, X_n)\right] \end{aligned}$$

where:

$$h(x_1, \ldots, x_n) = \mathbb{1}\{(x_1, \ldots, x_n) \in \Omega\} \cdot \varphi(x_1, \ldots, x_n)$$

Let $\hat{I}_{n_S}$ be the random variable defined by:

$$\hat{I}_{n_S} = \frac{1}{n_S} \sum_{s=1}^{n_S} h(X_{1,s}, \ldots, X_{n,s})$$

where $\{X_{1,s}, \ldots, X_{n,s}\}_{s \geq 1}$ is a sequence of *iid* random vectors with probability distribution $\mathcal{U}_{[\Omega]}$. Using the strong law of large numbers, we obtain:

$$\begin{aligned} \lim_{n_s \to \infty} \hat{I}_{n_s} &= \mathbb{E}\left[h\left(X_1, \ldots, X_n\right)\right] \\ &= \int \cdots \int_{\Omega} \varphi\left(x_1, \ldots, x_n\right) \mathrm{d}x_1 \cdots \mathrm{d}x_n \end{aligned}$$

Moreover, the central limit theorem states that:

$$\lim_{n_s \to \infty} \sqrt{n_S} \left(\frac{\hat{I}_{n_s} - I}{\sigma\left(h\left(X_1, \ldots, X_n\right)\right)} \right) = \mathcal{N}(0,1)$$

When n_S is large, we can deduce the following confidence interval:

$$\left[\hat{I}_{n_S} - c_\alpha \cdot \frac{\hat{S}_{n_S}}{\sqrt{n_S}}, \hat{I}_{n_S} + c_\alpha \cdot \frac{\hat{S}_{n_S}}{\sqrt{n_S}} \right]$$

where α is the confidence level, $c_\alpha = \Phi^{-1}\left((1+\alpha)/2\right)$ and $\hat{S}_{n_S}$ is the usual estimate of the standard deviation:

$$\hat{S}_{n_S} = \sqrt{\frac{1}{n_S - 1} \sum_{s=1}^{n_S} h^2\left(X_{1,s}, \ldots, X_{n,s}\right) - \hat{I}_{n_s}}$$

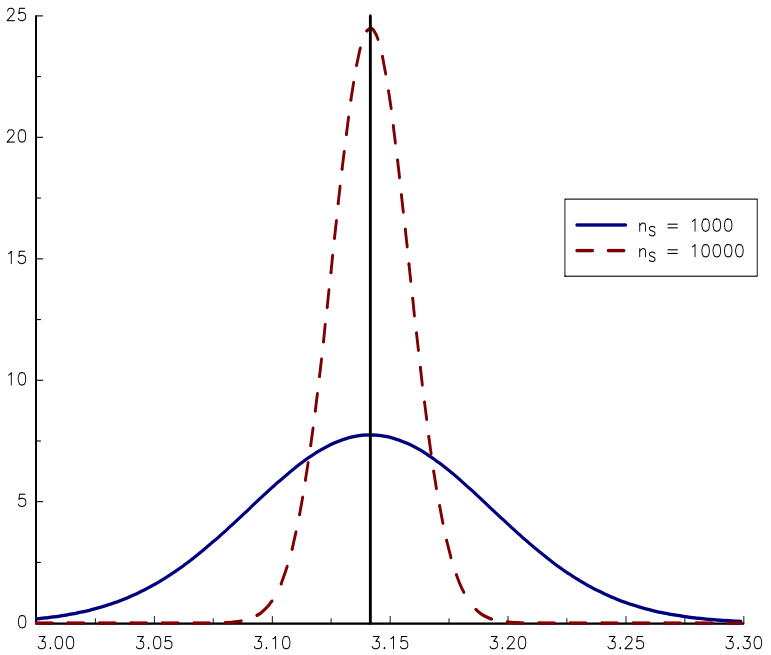

FIGURE 13.31: Density function of $\hat{\pi}_{n_S}$

We consider again the calculation of π. Previously, we obtain an estimate, which is far from the true value. In order to obtain a better precision, we can increase the number n_S

of simulations. In Figure 13.31, we report the density function of the estimator $\hat{\pi}_{n_S}$ when n_S is respectively equal to 1 000 and 10 000. We notice that the precision increases by a factor of $\sqrt{10}$ every time we multiply the number of simulations by ten. More generally, for a given precision p, we can deduce the sufficient number of simulations:

$$n_S \geq \left(c_\alpha \frac{\hat{S}_{n_S}}{p} \right)^2$$

In the case of the calculation of π, we have $\hat{S}_{n_S} \approx 1.642$. To obtain a precision of π with six digits after the decimal point at a 99% confidence level, we need about 18 trillion simulations:

$$\begin{aligned} n_S &\geq \left(\Phi^{-1}(0.995) \times \frac{1.642}{10^{-6}} \right)^2 \\ &\geq 17.9 \times 10^{12} \end{aligned}$$

Example 150 *We would like to calculate the following integral:*

$$I = \iiint_\Omega (x + y + z)^2 \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}z$$

This integral can be easily evaluated with Gaussian quadrature methods when Ω is a cube. However, the problem is more tricky when:

$$\Omega = \left\{ (x, y, z) \in \mathbb{R}_+^3 : x^2 + y^2 + z^2 \leq 25, x + y + z \geq 2 \right\}$$

Using the Monte Carlo method, we have $I = \mathbb{E}[h(X, Y, Z)]$ where X, Y and Z are three independent uniform random variables with probability distribution $\mathcal{U}_{[0,5]}$ and:

$$h(x, y, z) = \begin{cases} (x + y + z)^2 & \text{if} \quad (x, y, z) \in \Omega \\ 0 & \text{if} \quad (x, y, z) \notin \Omega \end{cases}$$

In Figure 13.32, we report the estimate $\hat{I}_{n_S}$ and the corresponding 99% confidence interval with respect to the number of simulations n_S.

13.3.1.3 Extension to the calculation of mathematical expectations

Let $X = (X_1, \ldots, X_n)$ be a random vector with probability distribution $\mathbf{F}$. We have:

$$\begin{aligned} \mathbb{E}[\varphi(X_1, \ldots, X_n)] &= \int \cdots \int \varphi(x_1, \ldots, x_n) \, \mathrm{d}\mathbf{F}(x_1, \cdots, x_n) \\ &= \int \cdots \int \varphi(x_1, \ldots, x_n) \, f(x_1, \cdots, x_n) \, \mathrm{d}x_1 \cdots \mathrm{d}x_n \\ &= \int \cdots \int h(x_1, \ldots, x_n) \, \mathrm{d}x_1 \cdots \mathrm{d}x_n \end{aligned}$$

where f is the density function. The Monte Carlo estimator of this integral is:

$$\hat{I}_{n_S} = \frac{1}{n_S} \sum_{s=1}^{n_S} \varphi(X_{1,s}, \ldots, X_{n,s})$$

where $\{X_{1,s}, \ldots, X_{n,s}\}_{s \geq 1}$ is a sequence of *iid* random vectors with probability distribution $\mathbf{F}$. Moreover, all the previous results hold in this general case where the random variables are not uniform.

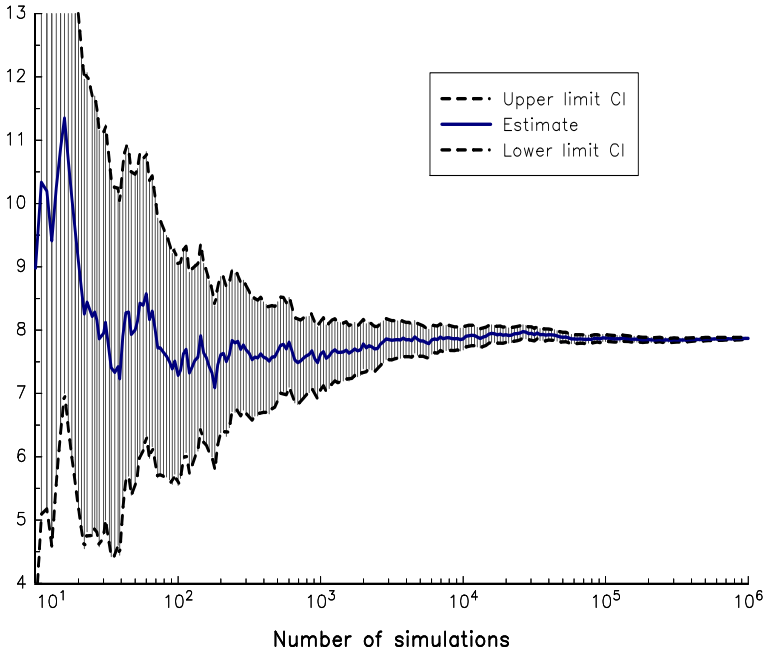

FIGURE 13.32: Convergence of the estimator $\hat{I}_{n_S}$

Example 151 *In the Black-Scholes model, the price of the look-back option with maturity T is given by:*

$$\mathcal{C} = e^{-rT}\mathbb{E}\left[\left(S(T) - \min_{0\leq t\leq T} S(t)\right)^+\right]$$

where the price $S(t)$ of the underlying asset is given by the following SDE:

$$\mathrm{d}S(t) = rS(t)\,\mathrm{d}t + \sigma S(t)\,\mathrm{d}W(t)$$

where r is the interest rate and σ is the volatility of the asset. It is difficult to calculate $\mathcal{C}$ analytically, because it requires the joint distribution of $S(T)$ and $\min_{0\leq t\leq T} S(t)$. However, we can easily calculate it using the Monte Carlo method. For a given simulation s, we use the exact scheme to simulate the geometric Brownian motion:

$$S_{m+1}^{(s)} = S_m^{(s)} \cdot \exp\left(\left(r - \frac{1}{2}\sigma^2\right)(t_{m+1} - t_m) + \sigma\sqrt{t_{m+1} - t_m}\cdot\varepsilon_m^{(s)}\right)$$

where $\varepsilon_m^{(s)} \sim \mathcal{N}(0,1)$ and $T = t_M$. The Monte Carlo estimator of the option price is then equal to:

$$\widehat{\mathcal{C}} = \frac{e^{-rT}}{n_S}\sum_{s=1}^{n_S}\left(S_M^{(s)} - \min_m S_m^{(s)}\right)^+$$

We deduce that the precision of the estimate depends on the number n_S of simulations, but also on the number M of discretization points. In Figure 13.33, the option price is calculated using these parameters: $S_0 = 100$, $r = 5\%$, $\sigma = 20\%$ and $T = 3/12$. We consider 100 000 simulations whereas the number M of discretization points varies between 5 and 100. We notice that the 99% confidence interval does not really depend on M. However, the option price increases with the number of discretization points. This is normal because

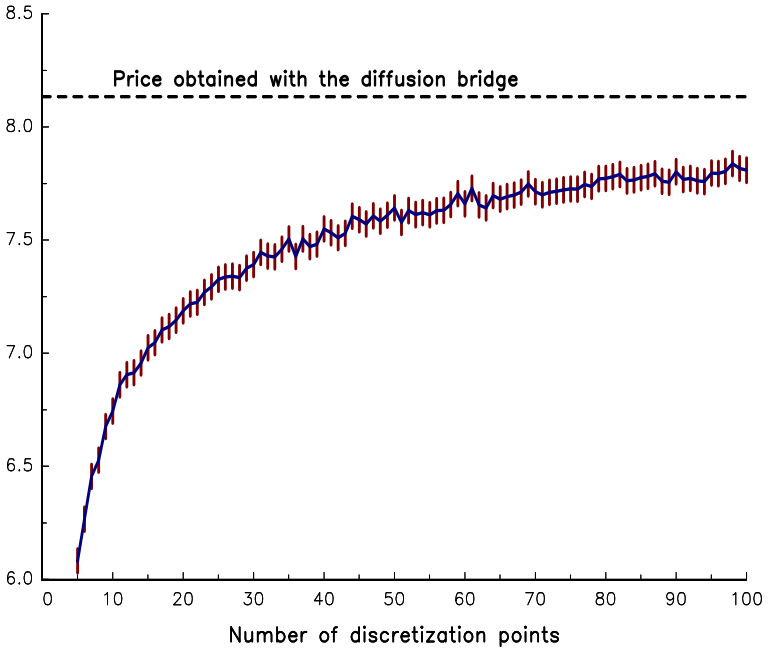

FIGURE 13.33: Computing the look-back option price

$\min_{0\leq t\leq T} S(t)$ *is always underestimated by* $\min_m S_m^{(s)}$. *This is why we have also reported the option price using the diffusion bridge approach. This example shows that the MC method does not always converge when the function* $\varphi(x_1,\ldots,x_n)$ *is approximated.*

Let us consider the following integral:

$$I = \int \cdots \int h(x_1,\ldots,x_n)\,\mathrm{d}x_1\cdots\mathrm{d}x_n$$

We can write it as follows:

$$I = \int \cdots \int \frac{h(x_1,\ldots,x_n)}{f(x_1,\cdots,x_n)} f(x_1,\cdots,x_n)\,\mathrm{d}x_1\cdots\mathrm{d}x_n$$

where $f(x_1,\cdots,x_n)$ is a multidimensional density function. We deduce that:

$$I = \mathbb{E}\left[\frac{h(X_1,\ldots,X_n)}{f(X_1,\ldots,X_n)}\right]$$

This implies that we can compute an integral with the MC method by using any multidimensional distribution function. If we apply this result to the calculation of π, we have:

$$\begin{aligned}
\pi &= \iint_{x^2+y^2\leq 1} \mathrm{d}x\,\mathrm{d}y \\
&= \iint \mathbb{1}\left\{x^2+y^2\leq 1\right\}\,\mathrm{d}x\,\mathrm{d}y \\
&= \iint \frac{\mathbb{1}\left\{x^2+y^2\leq 1\right\}}{\phi(x)\,\phi(y)}\phi(x)\,\phi(y)\,\mathrm{d}x\,\mathrm{d}y
\end{aligned}$$

We deduce that:

$$\pi = \mathbb{E}\left[\frac{\mathbb{1}\left\{X^2 + Y^2 \leq 1\right\}}{\phi(X)\,\phi(Y)}\right]$$

where X and Y are two independent standard Gaussian random variables. We can then estimate π by:

$$\hat{\pi}_{n_S} = \frac{1}{n_S}\sum_{s=1}^{n_S}\frac{\mathbb{1}\left\{x_s^2 + y_s^2 \leq 1\right\}}{\phi(x_s)\,\phi(y_s)}$$

where x_s and y_s are two independent random variates from the probability distribution $\mathcal{N}(0,1)$. For instance, we report the points (x_s, y_s) used to calculate π with 1 000 simulations in Figure 13.34.

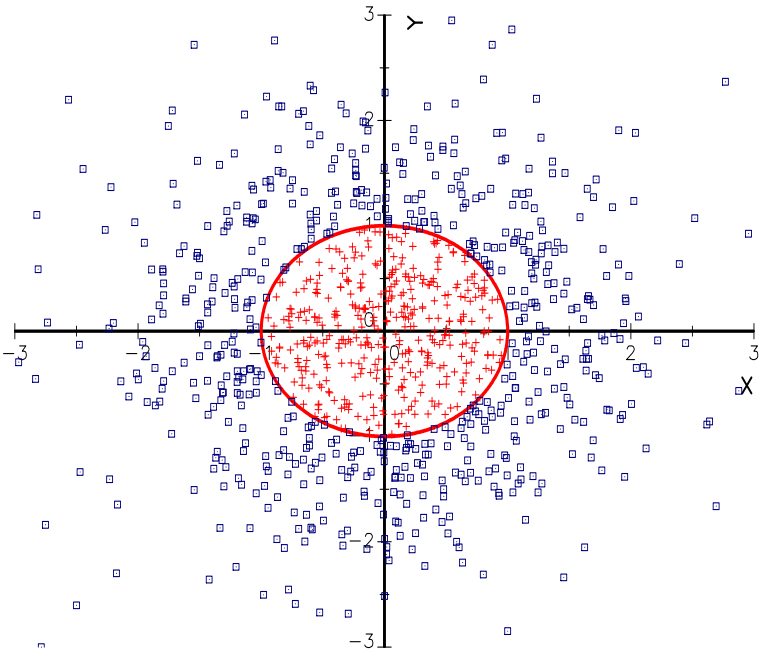

FIGURE 13.34: Computing π with normal random numbers

Remark 163 *The previous approach is particularly interesting when the set Ω is not bounded, which implies that we cannot use uniform random numbers.*

13.3.2 Variance reduction

We consider two unbiased estimators $\hat{I}_{n_S}^{(1)}$ and $\hat{I}_{n_S}^{(2)}$ of the integral I, meaning that $\mathbb{E}\left[\hat{I}_{n_S}^{(1)}\right] = \mathbb{E}\left[\hat{I}_{n_S}^{(2)}\right] = I$. We will say that $\hat{I}_{n_S}^{(1)}$ is more efficient than $\hat{I}_{n_S}^{(2)}$ if the inequality $\text{var}\left(\hat{I}_{n_S}^{(1)}\right) \leq \text{var}\left(\hat{I}_{n_S}^{(2)}\right)$ holds for all values of n_S that are larger than $n_S^\star$. Variance reduction is then the search of more efficient estimators.

13.3.2.1 Antithetic variates

Theoretical aspects We have:

$$I = \mathbb{E}\left[\varphi(X_1, \ldots, X_n)\right] = \mathbb{E}\left[Y\right]$$

where $Y = \varphi(X_1, \ldots, X_n)$ is a one-dimensional random variable. It follows that:

$$\hat{I}_{n_S} = \bar{Y}_{n_S} = \frac{1}{n_S} \sum_{s=1}^{n_S} Y_s$$

We now consider the estimators $\bar{Y}_{n_S}$ and $\bar{Y}'_{n_S}$ based on two different samples and define $\bar{Y}^\star$ as follows:

$$\bar{Y}^\star = \frac{\bar{Y}_{n_S} + \bar{Y}'_{n_S}}{2}$$

We have:

$$\begin{aligned}
\mathbb{E}\left[\bar{Y}^\star\right] &= \mathbb{E}\left[\frac{\bar{Y}_{n_S} + \bar{Y}'_{n_S}}{2}\right] \\
&= \mathbb{E}\left[\bar{Y}_{n_S}\right] \\
&= I
\end{aligned}$$

and:

$$\begin{aligned}
\text{var}\left(\bar{Y}^\star\right) &= \text{var}\left(\frac{\bar{Y}_{n_S} + \bar{Y}'_{n_S}}{2}\right) \\
&= \frac{1}{4}\text{var}\left(\bar{Y}_{n_S}\right) + \frac{1}{4}\text{var}\left(\bar{Y}'_{n_S}\right) + \frac{1}{2}\text{cov}\left(\bar{Y}_{n_S}, \bar{Y}'_{n_S}\right) \\
&= \frac{1 + \rho\left\langle \bar{Y}_{n_S}, \bar{Y}'_{n_S}\right\rangle}{2}\text{var}\left(\bar{Y}_{n_S}\right) \\
&= \frac{1 + \rho\left\langle Y_s, Y'_s\right\rangle}{2}\text{var}\left(\bar{Y}_{n_S}\right)
\end{aligned}$$

where[31] $\rho\left\langle Y_s, Y'_s\right\rangle$ is the correlation between Y_s and Y'_s. Because we have $\rho\left\langle Y_s, Y'_s\right\rangle \leq 1$, we deduce that:

$$\text{var}\left(\bar{Y}^\star\right) \leq \text{var}\left(\bar{Y}_{n_S}\right)$$

If we simulate the random variates Y_s and Y'_s independently, $\rho\left\langle Y_s, Y'_s\right\rangle$ is equal to zero and the variance of the estimator is divided by 2. However, the number of simulations have been multiplied by two. The efficiency of the estimator has then not been improved.

The underlying idea of antithetic variables is therefore to use two perfectly dependent random variables Y_s and Y'_s:

$$Y'_s = \psi(Y_s)$$

[31] We have:

$$\begin{aligned}
\text{cov}\left(\bar{Y}_{n_S}, \bar{Y}'_{n_S}\right) &= \mathbb{E}\left[\left(\frac{1}{n_S}\sum\nolimits_{s=1}^{n_S} Y_s - \mathbb{E}[Y]\right) \cdot \left(\frac{1}{n_S}\sum\nolimits_{s'=1}^{n_S} Y'_{s'} - \mathbb{E}[Y]\right)\right] \\
&= \frac{1}{n_S^2}\sum\nolimits_{s=1}^{n_S} \mathbb{E}\left[(Y_s - \mathbb{E}[Y]) \cdot \left(Y'_s - \mathbb{E}[Y]\right)\right] + \\
&\quad \frac{2}{n_S^2}\sum\nolimits_{s>s'}^{n_S} \mathbb{E}\left[(Y_s - \mathbb{E}[Y]) \cdot \left(Y'_{s'} - \mathbb{E}[Y]\right)\right] \\
&= \frac{1}{n_S} \cdot \text{cov}\left(Y_s, Y'_s\right) + \frac{2}{n_S^2} \cdot 0
\end{aligned}$$

It follows that:

$$\rho\left\langle \bar{Y}_{n_S}, \bar{Y}'_{n_S}\right\rangle = \frac{\text{cov}\left(\bar{Y}_{n_S}, \bar{Y}'_{n_S}\right)}{\sqrt{\text{var}\left(\bar{Y}_{n_S}\right) \cdot \text{var}\left(\bar{Y}'_{n_S}\right)}} = \rho\left(Y_s, Y'_s\right)$$

where ψ is a deterministic function. This implies that:

$$\bar{Y}_{n_S}^{\star} = \frac{1}{n_S}\sum_{s=1}^{n_S} Y_s^{\star}$$

where:

$$Y_s^{\star} = \frac{Y_s + Y_s'}{2} = \frac{Y_s + \psi(Y_s)}{2}$$

It follows that:

$$\rho\left\langle \bar{Y}_{n_S}, \bar{Y}_{n_S}' \right\rangle = \rho\left\langle Y, Y' \right\rangle = \rho\left\langle Y, \psi(Y) \right\rangle$$

Minimizing the variance $\text{var}\left(\bar{Y}^{\star}\right)$ is then equivalent to minimize the correlation $\rho\langle Y, \psi(Y)\rangle$. We also know that the correlation reaches its lower bound if the dependence function between Y and $\psi(Y)$ is equal to the lower Fréchet copula:

$$\mathbf{C}\langle Y, \psi(Y)\rangle = \mathbf{C}^{-}$$

However, $\rho\langle Y, \psi(Y)\rangle$ is not necessarily equal to -1 except in some special cases.

We consider the one-dimensional case with $Y = \varphi(X)$. If we assume that φ is an increasing function, it follows that:

$$\begin{aligned}\mathbf{C}\langle Y, \psi(Y)\rangle &= \mathbf{C}\langle \varphi(X), \psi(\varphi(X))\rangle \\ &= \mathbf{C}\langle X, \psi(X)\rangle\end{aligned}$$

To obtain the lower bound $\mathbf{C}^{-}$, X and $\psi(X)$ must be countermonotonic. We know that[32]:

$$\psi(X) = \mathbf{F}^{-1}(1 - \mathbf{F}(X)) \tag{13.8}$$

where $\mathbf{F}$ is the probability distribution of X. For instance, if $X \sim \mathcal{U}_{[0,1]}$, we have $X' = 1 - X$. In the case where $X \sim \mathcal{N}(0,1)$, we have:

$$\begin{aligned}X' &= \Phi^{-1}(1 - \Phi(X)) \\ &= \Phi^{-1}(\Phi(-X)) \\ &= -X\end{aligned}$$

Example 152 *We consider the following functions:*

1. $\varphi_1(x) = x^3 + x + 1$;
2. $\varphi_2(x) = x^4 + x^2 + 1$;
3. $\varphi_3(x) = x^4 + x^3 + x^2 + x + 1$;

For each function, we want to estimate $I = \mathbb{E}[\varphi(\mathcal{N}(0,1))]$ *using the antithetic estimator:*

$$\bar{Y}_{n_S}^{\star} = \frac{1}{n_S}\sum_{s=1}^{n_S} \frac{\varphi(X_s) + \varphi(-X_s)}{2}$$

where $X_s \sim \mathcal{N}(0,1)$. *We obtain the following results*[33]*:*

$\varphi(x)$	$\varphi_1(x)$	$\varphi_2(x)$	$\varphi_3(x)$
$\mathbb{E}[\varphi(X_s)]$ *or* $\mathbb{E}[\varphi(-X_s)]$	1	5	5
$\text{var}(\varphi(X_s))$ *or* $\text{var}(\varphi(-X_s))$	22	122	144
$\text{cov}(\varphi(X_s), \varphi(-X_s))$	-22	122	100
$\rho\langle\varphi(X_s), \varphi(-X_s)\rangle$	-1	1	$25/36$

[32] See Section 11.2.1 on page 722.

[33] Let $X \sim \mathcal{N}(0,1)$. We have $\mathbb{E}\left[X^2\right] = 1$, $\mathbb{E}\left[X^{2m}\right] = (2m-1)\,\mathbb{E}\left[X^{2m-2}\right]$ and $\mathbb{E}\left[X^{2m+1}\right] = 0$ for $m \in \mathbb{N}$.

We notice that the antithetic estimator is fully efficient in the first case, because its variance is equal to zero. In the second case, it is not efficient because we have $\text{var}\left(\bar{Y}_{n_S}^{\star}\right) = \text{var}\left(\bar{Y}_{n_S}\right)$. *Finally, the antithetic estimator reduces the variance by* 15.3% *in the last case.*

To understand these numerical results, we must study the relationship between $\mathbf{C}\langle X, X'\rangle$ and $\mathbf{C}\langle Y, Y'\rangle$. Indeed, we have:

$$\left\{\mathbf{C}\langle X, X'\rangle = \mathbf{C}^- \Rightarrow \mathbf{C}\langle Y, Y'\rangle = \mathbf{C}^-\right\} \Leftrightarrow \varphi'(x) \geq 0$$

We have represented the three functions $\varphi_1(x)$, $\varphi_2(x)$ and $\varphi_3(x)$ in Figure 13.35. Because $\varphi_1(x)$ is an increasing function, it follows that the copula function between Y and Y' reaches the lower Fréchet bound. The function $\varphi_2(x)$ is perfectly symmetric around $x = 0$. In this case, it is impossible to reduce the variance of the MC estimator by the use of antithetic Gaussian variates. Even if the function $\varphi_3(x)$ is not monotonous, it is however sufficiently asymmetric to obtain a low but significant reduction of the variance of the MC estimator.

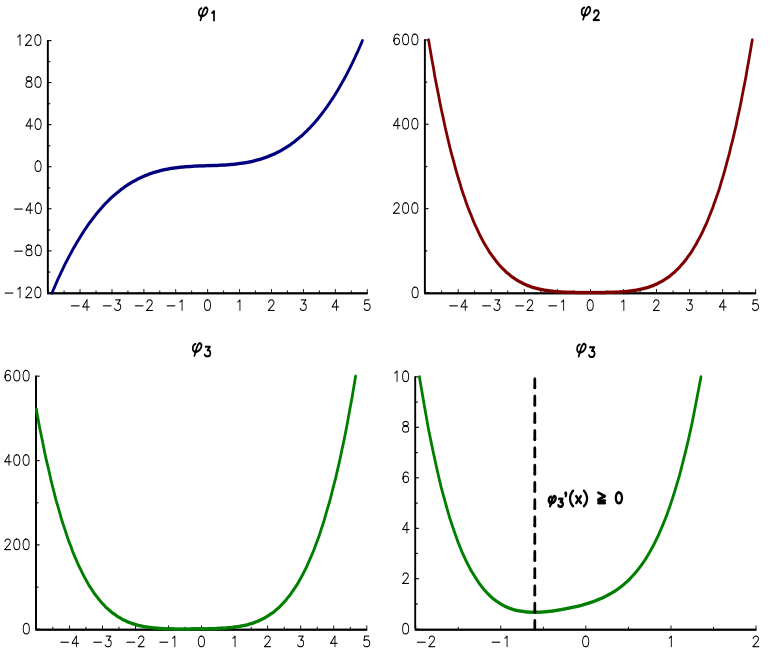

FIGURE 13.35: Functions $\varphi_1(x)$, $\varphi_2(x)$ and $\varphi_3(x)$

Remark 164 *In the case where* φ *is a decreasing function, we can show that the lower bound* $\mathbf{C}^-$ *between* Y *and* Y' *is also reached when* X *and* $\psi(X)$ *are countermonotonic (Ross, 2012).*

The extension of the previous results to the multidimensional case is not straightforward. Indeed, the copula condition between Y and Y' becomes:

$$\mathbf{C}\langle Y, Y'\rangle = \mathbf{C}^- \Leftrightarrow \mathbf{C}\langle \varphi(X_1, \ldots, X_n), \varphi(X_1', \ldots, X_n')\rangle = \mathbf{C}^-$$

where $X_1', \ldots, X_n'$ are the antithetic variates of $X_1, \ldots, X_n$. A natural generalization of the relationship (13.8) is:

$$X_i' = \mathbf{F}_i^{-1}(1 - \mathbf{F}_i(X_i))$$

where $\mathbf{F}_i$ is the probability distribution of X_i. By assuming that φ is a monotonic function of each of its arguments, Ross (2012) shows that:

$$\rho\left\langle Y, Y'\right\rangle < 0$$

where:

$$Y' = \varphi\left(\mathbf{F}_1^{-1}\left(1-\mathbf{F}_1\left(X_1\right)\right), \ldots, \mathbf{F}_n^{-1}\left(1-\mathbf{F}_n\left(X_n\right)\right)\right)$$

This means that we can reduce the variance of the MC estimator by using the antithetic variates $X_i' = \mathbf{F}_i^{-1}\left(1-\mathbf{F}_i\left(X_i\right)\right)$. However, it does not prove that this approach minimizes the correlation $\rho\left\langle Y, Y'\right\rangle$. Moreover, we have no results when φ is a general function.

Application to the geometric Brownian motion In the Gaussian case $X \sim \mathcal{N}(0,1)$, the antithetic variable is:

$$X' = -X$$

As the simulation of $Y \sim \mathcal{N}\left(\mu, \sigma^2\right)$ is obtained using the relationship $Y = \mu + \sigma X$, we deduce that the antithetic variable is:

$$\begin{aligned} Y' &= \mu - \sigma X \\ &= \mu - \sigma\frac{(Y-\mu)}{\sigma} \\ &= 2\mu - Y \end{aligned}$$

If we consider the geometric Brownian motion, the fixed-interval scheme is:

$$X_{m+1} = X_m \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)h + \sigma\sqrt{h}\cdot\varepsilon_m\right)$$

whereas the antithetic path is given by:

$$X'_{m+1} = X'_m \cdot \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right)h - \sigma\sqrt{h}\cdot\varepsilon_m\right)$$

In Figure 13.36, we report 4 trajectories of the GBM process and the corresponding antithetic paths[34].

In the multidimensional case, we recall that:

$$X_{j,m+1} = X_{j,m} \cdot \exp\left(\left(\mu_j - \frac{1}{2}\sigma_j^2\right)h + \sigma_j\sqrt{h}\cdot\varepsilon_{j,m}\right)$$

where $\varepsilon_m = (\varepsilon_{1,m}, \ldots, \varepsilon_{n,m}) \sim \mathcal{N}_n(\mathbf{0}, \rho)$. We simulate ε_m by using the relationship $\varepsilon_m = P \cdot \eta_m$ where $\eta_m \sim \mathcal{N}_n(\mathbf{0}, I_n)$ and P is the Cholesky matrix satisfying $PP^\top = \rho$. The antithetic trajectory is then:

$$X'_{j,m+1} = X'_{j,m} \cdot \exp\left(\left(\mu_j - \frac{1}{2}\sigma_j^2\right)h + \sigma_j\sqrt{k}\cdot\varepsilon'_{j,m}\right)$$

where:

$$\varepsilon'_m = -P \cdot \eta_m = -\varepsilon_m$$

We verify that $\varepsilon'_m = \left(\varepsilon'_{1,m}, \ldots, \varepsilon'_{n,m}\right) \sim \mathcal{N}_n(\mathbf{0}, \rho)$.

[34] The parameter values are $X_0 = 100$, $\mu = 10\%$ and $\sigma = 20\%$.

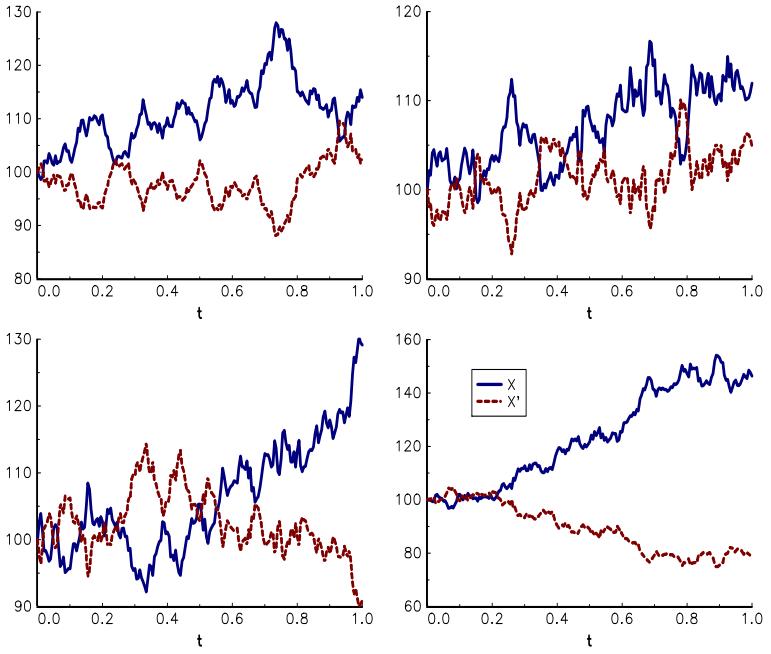

FIGURE 13.36: Antithetic simulation of the GBM process

Example 153 *In the Black-Scholes model, the price of the spread option with maturity T and strike K is given by:*

$$\mathcal{C} = e^{-rT}\mathbb{E}\left[\left(S_1(T) - S_2(T) - K\right)^+\right]$$

where the prices $S_1(t)$ and $S_2(t)$ of the underlying assets are given by the following SDE:

$$\begin{cases} \mathrm{d}S_1(t) = rS_1(t)\,\mathrm{d}t + \sigma_1 S_1(t)\,\mathrm{d}W_1(t) \\ \mathrm{d}S_2(t) = rS_2(t)\,\mathrm{d}t + \sigma_2 S_2(t)\,\mathrm{d}W_2(t) \end{cases}$$

and $\mathbb{E}[W_1(t)\,W_2(t)] = \rho t$. To calculate the option price using Monte Carlo methods, we simulate the bivariate GBM $S_1(t)$ and $S_2(t)$ and the MC estimator is:

$$\hat{\mathcal{C}}_{\mathrm{MC}} = \frac{e^{-rT}}{n_S}\sum_{s=1}^{n_S}\left(S_1^{(s)}(T) - S_2^{(s)}(T) - K\right)^+$$

where $S_j^{(s)}(T)$ is the s^{th} simulation of the terminal value $S_j(T)$. For the AV estimator, we obtain:

$$\hat{\mathcal{C}}_{\mathrm{AV}} = \frac{e^{-rT}}{n_S}\sum_{s=1}^{n_S}\frac{\left(S_1^{(s)}(T) - S_2^{(s)}(T) - K\right)^+ + \left(S_1^{\prime(s)}(T) - S_2^{\prime(s)}(T) - K\right)^+}{2}$$

where $S_j^{\prime(s)}(T)$ is the antithetic variate of $S_j^{(s)}(T)$. In Figure 13.37, we report the probability density function of the estimators $\hat{\mathcal{C}}_{\mathrm{MC}}$ and $\hat{\mathcal{C}}_{\mathrm{AV}}$ when n_S is equal to 1 000[35]*. We observe*

[35]The parameters are $S_1(0) = S_2(0) = 100$, $r = 5\%$, $\sigma_1 = \sigma_2 = 20\%$, $\rho = 50\%$, $T = 1$ and $K = 5$.

that the variance reduction is significative and we obtain:

$$\frac{\text{var}\left(\hat{\mathcal{C}}_{\text{AV}}\right)}{\text{var}\left(\hat{\mathcal{C}}_{\text{MC}}\right)} = 34.7\%$$

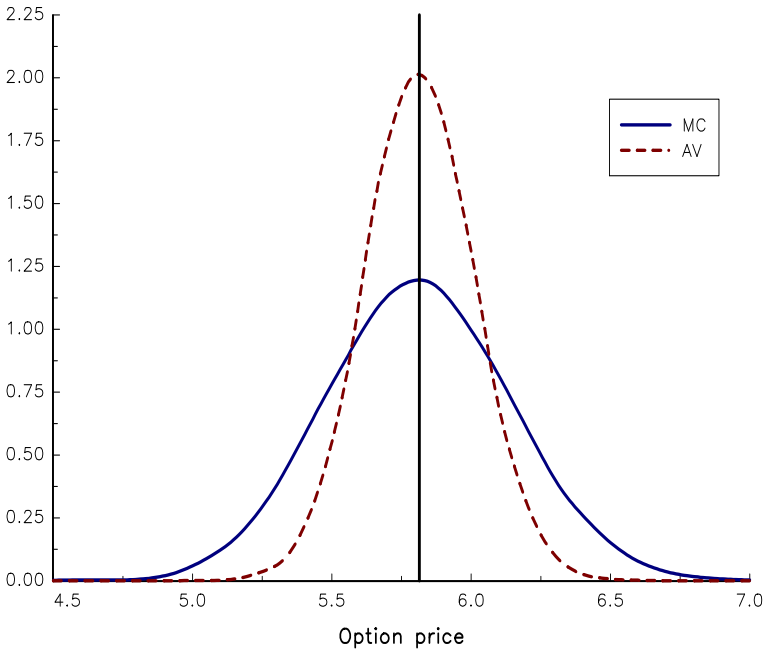

FIGURE 13.37: Probability density function of $\widehat{\mathcal{C}}_{\text{MC}}$ and $\widehat{\mathcal{C}}_{\text{AV}}$ ($n_S = 1\,000$)

13.3.2.2 Control variates

Let $Y = \varphi(X_1, \ldots, X_n)$ and V be a random variable with known mean $\mathbb{E}[V]$. We define Z as follows:

$$Z = Y + c \cdot (V - \mathbb{E}[V])$$

We deduce that:

$$\begin{aligned}
\mathbb{E}[Z] &= \mathbb{E}[Y + c \cdot (V - \mathbb{E}[V])] \\
&= \mathbb{E}[Y] + c \cdot \mathbb{E}[V - \mathbb{E}[V]] \\
&= \mathbb{E}[\varphi(X_1, \ldots, X_n)]
\end{aligned}$$

and:

$$\begin{aligned}
\text{var}(Z) &= \text{var}(Y + c \cdot (V - \mathbb{E}[V])) \\
&= \text{var}(Y) + 2 \cdot c \cdot \text{cov}(Y, V) + c^2 \cdot \text{var}(V)
\end{aligned}$$

It follows that:

$$\begin{aligned}
\text{var}(Z) \leq \text{var}(Y) &\Leftrightarrow 2 \cdot c \cdot \text{cov}(Y, V) + c^2 \cdot \text{var}(V) \leq 0 \\
&\Rightarrow c \cdot \text{cov}(Y, V) \leq 0
\end{aligned}$$

In order to obtain a lower variance, a necessary condition is that c and $\text{cov}(Y,V)$ have opposite signs. The minimum is obtained when $\partial_c \text{var}(Z) = 0$ or equivalently when:

$$c^\star = -\frac{\text{cov}(Y,V)}{\text{var}(V)} = -\beta$$

The optimal value $c^\star$ is then equal to the opposite of the beta of Y with respect to the control variate V. In this case, we have:

$$Z = Y - \frac{\text{cov}(Y,V)}{\text{var}(V)} \cdot (V - \mathbb{E}[V])$$

and:

$$\begin{aligned}\text{var}(Z) &= \text{var}(Y) - \frac{\text{cov}^2(Y,V)}{\text{var}(V)} \\ &= \left(1 - \rho^2 \langle Y,V \rangle\right) \cdot \text{var}(Y)\end{aligned}$$

This implies that we have to choose a control variate V that is highly (positively or negatively) correlated with Y in order to reduce the variance.

Example 154 *We consider that* $X \sim \mathcal{U}_{[0,1]}$ *and* $\varphi(x) = e^x$. *We would like to estimate:*

$$I = \mathbb{E}[\varphi(X)] = \int_0^1 e^x \,\mathrm{d}x$$

We set $Y = e^X$ *and* $V = X$. *We know that* $\mathbb{E}[V] = 1/2$ *and* $\text{var}(V) = 1/12$. *It follows that:*

$$\begin{aligned}\text{var}(Y) &= \mathbb{E}\left[Y^2\right] - \mathbb{E}^2[Y] \\ &= \int_0^1 e^{2x} \,\mathrm{d}x - \left(\int_0^1 e^x \,\mathrm{d}x\right)^2 \\ &= \left[\frac{e^{2x}}{2}\right]_0^1 - \left(e^1 - e^0\right)^2 \\ &= \frac{4e - e^2 - 3}{2} \\ &\approx 0.2420\end{aligned}$$

and:

$$\begin{aligned}\text{cov}(Y,V) &= \mathbb{E}[VY] - \mathbb{E}[V]\,\mathbb{E}[Y] \\ &= \int_0^1 x e^x \,\mathrm{d}x - \frac{1}{2}\left(e^1 - e^0\right) \\ &= \left[x e^x\right]_0^1 - \int_0^1 e^x \,\mathrm{d}x - \frac{1}{2}\left(e^1 - e^0\right) \\ &= \frac{3 - e}{2} \\ &\approx 0.1409\end{aligned}$$

If we consider the VC estimator Z *defined by*[36]*:*

$$\begin{aligned}Z &= Y - \frac{\text{cov}(Y,V)}{\text{var}(V)} \cdot (V - \mathbb{E}[V]) \\ &= Y - (18 - 6e) \cdot \left(V - \frac{1}{2}\right)\end{aligned}$$

[36] We have $\beta \approx 1.6903$.

we obtain:

$$\begin{aligned}\text{var}(Z) &= \text{var}(Y) - \frac{\text{cov}^2(Y,V)}{\text{var}(V)} \\ &= \frac{4e - e^2 - 3}{2} - 3 \cdot (3-e)^2 \\ &\approx 0.0039\end{aligned}$$

We conclude that we have dramatically reduced the variance of the estimator, because we have:

$$\frac{\text{var}\left(\hat{I}_{\text{CV}}\right)}{\text{var}\left(\hat{I}_{\text{MC}}\right)} = \frac{\text{var}(Z)}{\text{var}(Y)} = 1.628\%$$

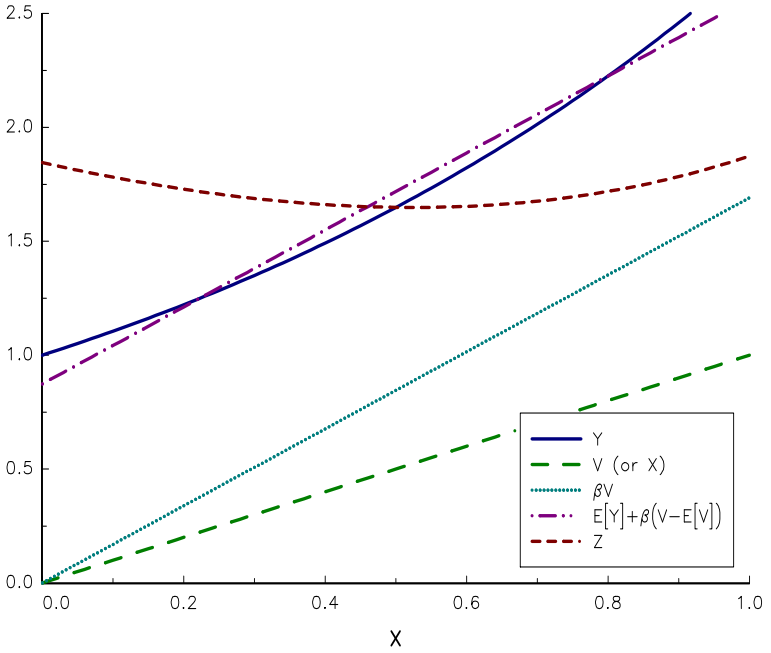

FIGURE 13.38: Understanding the variance reduction in control variates

This example may be disturbing, because the variance reduction is huge. To understand the mechanisms underlying control variates, we illustrate the previous example in Figure 13.38. For each variable, we have represented the relationship with respect to the random variable X. We have $Y = \exp(X)$ and $V = X$. To maximize the dependence between Y and the control variate, it is better to consider βV instead of V. However, the random variable βV is not well located, because it does not fit well Y. This is not the case of $\hat{Y} = \mathbb{E}[Y] + \beta(V - \mathbf{E}[V])$. Indeed, $\hat{Y}$ is the conditional expectation of Y with respect to V:

$$\mathbb{E}[Y \mid V] = \mathbb{E}[Y] + \beta(V - \mathbf{E}[V])$$

This is the best linear estimator of Y. The residual U of the linear regression is then equal to:

$$\begin{aligned}U &= Y - \hat{Y} \\ &= (Y - \mathbb{E}[Y]) - \beta(V - \mathbf{E}[V])\end{aligned}$$

The CV estimator Z is a translation of the residual in order to satisfy $\mathbb{E}[Z] = \mathbb{E}[Y]$:

$$\begin{aligned} Z &= \mathbb{E}[Y] + U \\ &= Y - \beta(V - \mathbf{E}[V]) \end{aligned}$$

By construction, the variance of the residual U is lower than the variance of the random variable Y. We conclude that:

$$\operatorname{var}(Z) = \operatorname{var}(U) \leq \operatorname{var}(Y)$$

We can therefore obtain a large variance reduction if the following conditions are satisfied:

- the control variate V largely explains the random variable Y;
- the relationship between Y and V is almost linear.

In the previous example, these conditions are largely satisfied and the residuals are very small[37].

Remark 165 *In practice, we don't know the optimal value $c^\star$. However, the previous framework helps us to estimate it. Indeed, we have:*

$$c^\star = -\hat{\beta}$$

where $\hat{\beta}$ is the OLS estimate associated to the linear regression model:

$$Y_s = \alpha + \beta V_s + u_s$$

Because Y_s and V_s are the simulated values of Y and V, this implies that $c^\star$ is calculated at the final step of the Monte Carlo method.

We recall that the price of an arithmetic Asian call option is given by:

$$\mathcal{C} = e^{-rT}\mathbb{E}\left[\left(\bar{S} - K\right)^+\right]$$

where K is the strike of the option and $\bar{S}$ denotes the average of $S(t)$ on a given number of fixing dates[38] $\{t_1, \ldots, t_{n_F}\}$:

$$\bar{S} = \frac{1}{n_F}\sum_{m=1}^{n_F} S(t_m)$$

We can estimate the option price using the Black-Scholes model. We can also reduce the variance of the MC estimator by considering the following control variates:

1. the terminal value $V_1 = S(T)$ of the underlying asset;
2. the average value $V_2 = \bar{S}$;
3. the discounted payoff of the call option $V_3 = e^{-rT}(S(T) - K)^+$;
4. the discounted payoff of the geometric Asian call option $V_4 = e^{-rT}\left(\tilde{S} - K\right)^+$ where:

$$\tilde{S} = \left(\prod_{m=1}^{n_F} S(t_m)\right)^{1/n_F}$$

[37] The variance of residuals represents 1.628% of the variance of Y.
[38] We have $t_{n_F} = T$.

For these control variates, we know the expected value. In the first and second cases, we have:

$$\mathbb{E}\left[S\left(T\right)\right] = S_0 e^{rT}$$

and:

$$\mathbb{E}\left[\bar{S}\right] = \frac{S_0}{n_F}\sum_{m=1}^{n_F} e^{rt_m}$$

The expected value of the third control variate is the Black-Scholes formula of the European call option. For the last control variate, we have:

$$\begin{aligned}\tilde{S} &= \left(\prod_{m=1}^{n_F} S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)t_m+\sigma W(t_m)}\right)^{1/n_F} \\ &= S_0 \cdot \exp\left(\left(r-\frac{1}{2}\sigma^2\right)\bar{t}+\sigma\bar{W}\right)\end{aligned}$$

where:

$$\bar{t} = \frac{1}{n_F}\sum\nolimits_{m=1}^{n_F} t_m$$

and:

$$\bar{W} = \frac{1}{n_F}\sum\nolimits_{m=1}^{n_F} W\left(t_m\right)$$

Because $\tilde{S}$ has a log-normal distribution, we deduce that the expected value of the fourth control variate is also given by a Black-Scholes formula[39]. We consider the following parameters $S_0 = 100$, $K = 104$, $r = 5\%$, $\sigma = 20\%$ and $T = 5$. The fixing dates of the Asian option are $t_1 = 1$, $t_2 = 2$, $t_3 = 3$, $t_4 = 4$ and $t_5 = 5$. In top panels in Figure 13.39, we report the probability density function of the MC estimator $\hat{\boldsymbol{\mathcal{C}}}_{\text{MC}}$ and the CV estimator $\hat{\boldsymbol{\mathcal{C}}}_{\text{CV}}$ when the number of simulations is equal to 1 000. The variance ratio $\text{var}\left(\hat{\boldsymbol{\mathcal{C}}}_{\text{CV}}\right) / \text{var}\left(\hat{\boldsymbol{\mathcal{C}}}_{\text{MC}}\right)$ is respectively equal to 22.6% for $V_1 = S\left(T\right)$, 9.4% for $V_2 = \bar{S}$, 19.5% for $V_3 = e^{-rT}\left(S\left(T\right)-K\right)^+$ and 0.5% for $V_4 = e^{-rT}\left(\tilde{S}-K\right)^+$. In bottom panels in Figure 13.39, we also show the relationship between the simulated value $Y = e^{-rT}\left(\bar{S}-K\right)^+$ and the control variates V_1 and V_4. We verify that the linear regression produces lower residuals for V_4 than for V_1.

[39] We have:

$$\mathbb{E}\left[\ln\tilde{S}\right] = \ln S_0 + \left(r-\frac{1}{2}\sigma^2\right)\bar{t}$$

and:

$$\text{var}\left(\ln\tilde{S}\right) = \sigma^2\upsilon$$

where:

$$\begin{aligned}\upsilon &= \text{var}\left(\frac{1}{n_F}\sum\nolimits_{m=1}^{n_F} W\left(t_m\right)\right) \\ &= \frac{1}{n_F^2}\mathbb{E}\left[\sum\nolimits_{m=1}^{n_F} W^2\left(t_m\right) + 2\sum\nolimits_{m'>m} W\left(t_{m'}\right)W\left(t_m\right)\right] \\ &= \frac{1}{n_F^2}\left(\sum\nolimits_{m=1}^{n_F} t_m + 2\sum\nolimits_{m=1}^{n_F}\left(n_F-m\right)t_m\right)\end{aligned}$$

We deduce that:

$$\mathbb{E}\left[e^{-rT}\left(\tilde{S}-K\right)^+\right] = S_0 e^{\gamma-rT}\Phi\left(d+\sigma\sqrt{\upsilon}\right) - Ke^{-rT}\Phi\left(d\right)$$

where:

$$d = \frac{1}{\sigma\sqrt{\upsilon}}\left(\ln\frac{S_0}{K} + \left(r-\frac{1}{2}\sigma^2\right)\bar{t}\right)$$

and:

$$\gamma = r\bar{t} + \frac{1}{2}\sigma^2\left(\upsilon-\bar{t}\right)$$

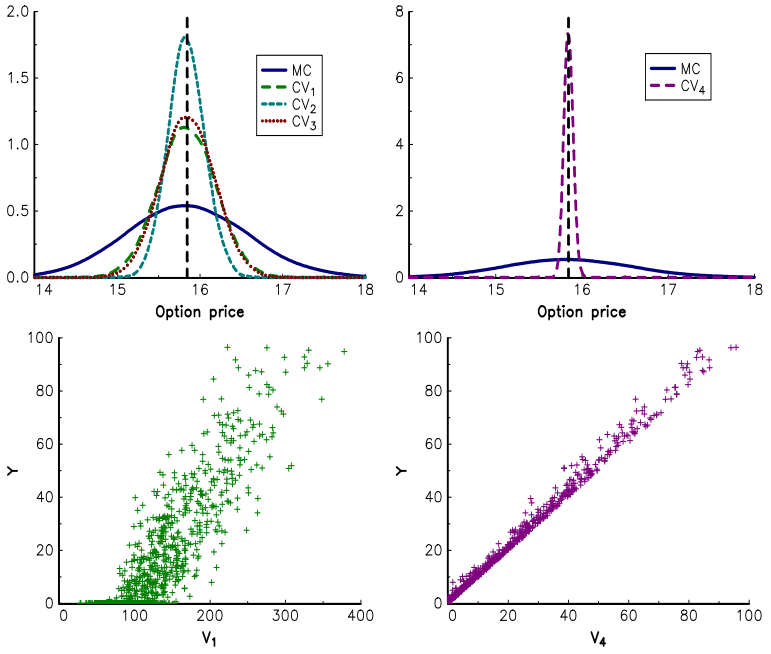

FIGURE 13.39: CV estimator of the arithmetic Asian call option

The previous approach can be extended in the case of several control variates:

$$\begin{aligned} Z &= Y + \sum_{i=1}^{n_{CV}} c_i \cdot (V_i - \mathbb{E}[V_i]) \\ &= Y + c^\top (V - \mathbb{E}[V]) \end{aligned}$$

where $c = (c_1, \ldots, c_{n_{CV}})$ and $V = (V_1, \ldots, V_{n_{CV}})$. We can show that the optimal value of c is equal to:

$$c^\star = -\operatorname{cov}(V, V)^{-1} \cdot \operatorname{cov}(V, Y)$$

By noting that minimizing the variance of Z is equivalent to minimize the variance of U where:

$$\begin{aligned} U &= Y - \hat{Y} \\ &= Y - \left(\alpha + \beta^\top V\right) \end{aligned}$$

we deduce that $c^\star = -\beta$. It follows that

$$\begin{aligned} \operatorname{var}(Z) &= \operatorname{var}(U) \\ &= \left(1 - R^2\right) \cdot \operatorname{var}(Y) \end{aligned}$$

where R^2 is the R-squared coefficient of the linear regression $Y = \alpha + \beta^\top V + U$.

Let us consider the previous example of the arithmetic Asian call option. In Table 13.5, we give the results of the linear regression by considering the combination of the four control variates. Previously, we found that the variance ratio was equal to 9.4% for the second control variate. If we combine the first three variates, this ratio becomes 3.5%. With the four control variates, the variance of the Monte Carlo estimator is divided by a factor of 500!

TABLE 13.5: Linear regression between the Asian call option and the control variates

$\hat{\alpha}$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	R^2	$1-R^2$
−51.482	0.036	0.538			90.7%	9.3%
−24.025	−0.346	0.595	0.548		96.5%	3.5%
−4.141	0.069		0.410		81.1%	18.9%
−38.727		0.428	0.174		92.9%	7.1%
−1.559	−0.040	0.054	0.111	0.905	99.8%	0.2%

Remark 166 *The reader may consult the book of Lamberton and Lapeyre (2007) for other examples of control variates in option pricing. In particular, they show how to use the put-call parity formula for reducing the volatility by noting that the variance of put options are generally smaller than the variance of call options.*

13.3.2.3 Importance sampling

Let $X = (X_1, \ldots, X_n)$ be a random vector with distribution function $\mathbf{F}$. We have:

$$\begin{aligned} I &= \mathbb{E}\left[\varphi\left(X_1, \ldots, X_n\right) \mid \mathbf{F}\right] \\ &= \int \cdots \int \varphi\left(x_1, \ldots, x_n\right) f\left(x_1, \ldots, x_n\right) \mathrm{d}x_1 \cdots \mathrm{d}x_n \end{aligned}$$

where $f\left(x_1, \ldots, x_n\right)$ is the probability density function of X. It follows that:

$$\begin{aligned} I &= \int \cdots \int \left(\varphi\left(x_1, \ldots, x_n\right) \frac{f\left(x_1, \ldots, x_n\right)}{g\left(x_1, \ldots, x_n\right)}\right) g\left(x_1, \ldots, x_n\right) \mathrm{d}x_1 \cdots \mathrm{d}x_n \\ &= \mathbb{E}\left[\varphi\left(X_1, \ldots, X_n\right) \frac{f\left(X_1, \ldots, X_n\right)}{g\left(X_1, \ldots, X_n\right)} \middle| \mathbf{G}\right] \\ &= \mathbb{E}\left[\varphi\left(X_1, \ldots, X_n\right) \mathcal{L}\left(X_1, \ldots, X_n\right) \mid \mathbf{G}\right] \end{aligned} \tag{13.9}$$

where $g\left(x_1, \ldots, x_n\right)$ is the probability density function of $\mathbf{G}$ and $\mathcal{L}$ is the likelihood ratio:

$$\mathcal{L}\left(x_1, \ldots, x_n\right) = \frac{f\left(x_1, \ldots, x_n\right)}{g\left(x_1, \ldots, x_n\right)}$$

The values taken by $\mathcal{L}\left(x_1, \ldots, x_n\right)$ are also called the importance sampling weights. Using the vector notation, the relationship (13.9) becomes:

$$\mathbb{E}\left[\varphi\left(X\right) \mid \mathbf{F}\right] = \mathbb{E}\left[\varphi\left(X\right) \mathcal{L}\left(X\right) \mid \mathbf{G}\right]$$

It follows that:

$$\mathbb{E}\left[\hat{I}_{\mathrm{MC}}\right] = \mathbb{E}\left[\hat{I}_{\mathrm{IS}}\right] = I$$

where $\hat{I}_{\mathrm{MC}}$ and $\hat{I}_{\mathrm{IS}}$ are the Monte Carlo and importance sampling estimators of I. We also deduce that[40]:

$$\mathrm{var}\left(\hat{I}_{\mathrm{IS}}\right) = \mathrm{var}\left(\varphi\left(X\right) \mathcal{L}\left(X\right) \mid \mathbf{G}\right)$$

[40] Recall that we use the vector notation, meaning that $x = (x_1, \ldots, x_n)$.

It follows that:

$$\begin{aligned}
\operatorname{var}\left(\hat{I}_{\mathrm{IS}}\right) &= \mathbb{E}\left[\varphi^2(X)\mathcal{L}^2(X) \mid \mathbf{G}\right] - \mathbb{E}^2\left[\varphi(X)\mathcal{L}(X) \mid \mathbf{G}\right] \\
&= \int \varphi^2(x)\mathcal{L}^2(x)g(x)\,\mathrm{d}x - I^2 \\
&= \int \varphi^2(x)\frac{f^2(x)}{g^2(x)}g(x)\,\mathrm{d}x - I^2 \\
&= \int \varphi^2(x)\frac{f^2(x)}{g(x)}\,\mathrm{d}x - I^2
\end{aligned} \tag{13.10}$$

If we compare the variance of the two estimators $\hat{I}_{\mathrm{MC}}$ and $\hat{I}_{\mathrm{IS}}$, we obtain:

$$\begin{aligned}
\operatorname{var}\left(\hat{I}_{\mathrm{IS}}\right) - \operatorname{var}\left(\hat{I}_{\mathrm{MC}}\right) &= \int \varphi^2(x)\frac{f^2(x)}{g(x)}\,\mathrm{d}x - \int \varphi^2(x)f(x)\,\mathrm{d}x \\
&= \int \varphi^2(x)\left(\frac{f(x)}{g(x)} - 1\right)f(x)\,\mathrm{d}x \\
&= \int \varphi^2(x)\left(\mathcal{L}(x) - 1\right)f(x)\,\mathrm{d}x
\end{aligned}$$

The difference may be negative if the weights $\mathcal{L}(x)$ are small ($\mathcal{L}(x) \ll 1$) because the values of $\varphi^2(x)f(x)$ are positive. The importance sampling approach changes then the importance of some values x by transforming the original probability distribution $\mathbf{F}$ into another probability distribution $\mathbf{G}$. Equation (13.10) is also interesting because it gives us some insights about the optimal IS distribution[41]:

$$\begin{aligned}
g^\star(x) &= \arg\min \operatorname{var}\left(\hat{I}_{\mathrm{IS}}\right) \\
&= \arg\min \int \varphi^2(x)\frac{f^2(x)}{g(x)}\,\mathrm{d}x \\
&= c \cdot |\varphi(x)| \cdot f(x)
\end{aligned}$$

where c is the normalizing constant such that $\int g^\star(x)\,\mathrm{d}x = 1$. A good choice of the IS density $g(x)$ is then an approximation of $|\varphi(x)| \cdot f(x)$ such that $g(x)$ can easily be simulated.

Remark 167 *In order to simplify the notation and avoid confusions, we consider that* $X \sim \mathbf{F}$ *and* $Z \sim \mathbf{G}$ *in the sequel. This means that* $\hat{I}_{\mathrm{MC}} = \varphi(X)$ *and* $\hat{I}_{\mathrm{IS}} = \varphi(Z)\mathcal{L}(Z)$.

We consider the estimation of the probability $p = \Pr\{X \geq 3\}$ when $X \sim \mathcal{N}(0,1)$. We have:

$$\varphi(x) = \mathbb{1}\{x \geq 3\}$$

Because the probability p is low ($\Pr\{X \geq 3\} \approx 0.1350\%$), the MC estimator will not be efficient. Indeed, it will be rare to simulate a random variate greater than 3. To reduce the variance of the MC estimator, we can use important sampling with $Z \sim \mathcal{N}\left(\mu_z, \sigma_z^2\right)$. For $\mu_z = 3$ and $\sigma_z = 1$, we report in Figure 13.40 the histogram of the estimators[42] $\hat{p}_{\mathrm{MC}}$ and

[41] The first-order condition is:

$$-\varphi^2(x) \cdot \frac{f^2(x)}{g^2(x)} = \lambda$$

where λ is a constant.

[42] We have:

$$\hat{p}_{\mathrm{MC}} = \frac{1}{n_S}\sum_{s=1}^{n_S} \mathbb{1}\{X_s \geq 3\}$$

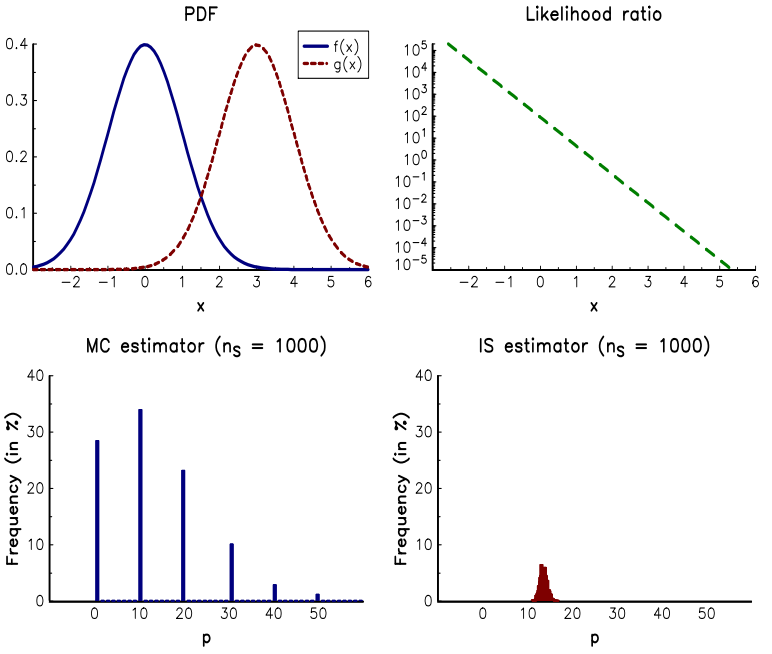

FIGURE 13.40: Histogram of the MC and IS estimators ($n_S = 1\,000$)

$\hat{p}_{\text{IS}}$ when the number of simulations is equal to 1 000. It is obvious that the IS estimator is better than the MC estimator. To explain that, we report the probability density function of X and Z in the top/left panel in Figure 13.40. Whereas $\Pr\{X \geq 3\}$ is close to zero, the probability $\Pr\{Z \geq 3\}$ is equal to 50%. Therefore, it is easier to simulate $Z \geq 3$, but we have to apply a correction to obtain the right probability. This correction is given by the likelihood ratio, which is represented in the top/right panel. In Figure 13.41, we show the standard deviation $\sigma(\hat{p}_{\text{IS}})$ for different values of μ_z and σ_z. When $\sigma_z = 1$ and $\mu_z \in [0, 5]$, it is lower than the standard deviation of $\hat{p}_{\text{MC}}$. For $\mu_z = 3$, the variance ratio is approximately equal to 1% meaning that the variance of $\hat{p}_{\text{MC}}$ is divided by a factor of 100. We also notice that we reduce the variance by using a higher value of σ_z. In fact, we can anticipate that the IS estimator is more efficient than the MC estimator if the following condition holds:

$$\Pr\{Z \geq 3\} \geq \Pr\{X \geq 3\}$$

The calculation of the optimal values of μ_z and σ_z is derived in Exercise 13.4.9 on page 891.

---

and:

$$\hat{p}_{\text{IS}} = \frac{1}{n_S} \sum_{s=1}^{n_S} \mathbb{1}\{Z_s \geq 3\} \cdot \mathcal{L}(Z_s)$$

where:

$$\mathcal{L}(z) = \sigma_z \exp\left(\frac{1}{2}\left(\frac{z - \mu_z}{\sigma_z}\right)^2 - \frac{1}{2}z^2\right)$$

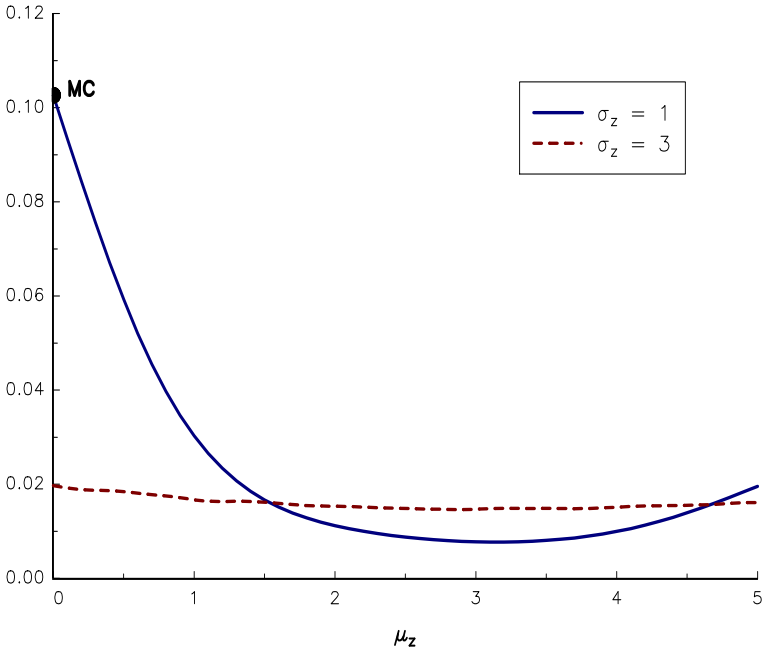

FIGURE 13.41: Standard deviation (in %) of the estimator $\hat{p}_{\text{IS}}$ ($n_S = 1\,000$)

Remark 168 *The previous example is an illustration of rare event simulation. This is why importance sampling is related to the theory of large deviations. Many results of this statistical field (Cramer's theorem, Berry-Esseen bounds) are then obtained using the same approach than the importance sampling method.*

We consider the pricing of the put option:

$$\boldsymbol{\mathcal{P}} = e^{-rT}\mathbb{E}\left[(K - S(T))^+\right]$$

We can estimate the option price by using the Monte Carlo method with:

$$\varphi(x) = e^{-rT}(K - x)^+$$

In the case where $K \ll S(0)$, the probability of exercise $\Pr\{S(T) \leq K\}$ is very small. Therefore, we have to increase the probability of exercise in order to obtain a more efficient estimator. In the case of the Black-Scholes model, the density function of $S(T)$ is equal to:

$$f(x) = \frac{1}{x\sigma_x}\phi\left(\frac{\ln x - \mu_x}{\sigma_x}\right)$$

where $\mu_x = \ln S_0 + \left(r - \sigma^2/2\right)T$ and $\sigma_x = \sigma\sqrt{T}$. Using the same approach than previously, we consider the IS density $g(x)$ defined by:

$$g(x) = \frac{1}{x\sigma_z}\phi\left(\frac{\ln x - \mu_z}{\sigma_z}\right)$$

where $\mu_z = \theta + \mu_x$ and $\sigma_z = \sigma_x$. For instance, we can choose θ such that the probability of exercise is equal to 50%. It follows that:

$$\begin{aligned}\Pr\{Z \leq K\} = \frac{1}{2} &\Leftrightarrow \Phi\left(\frac{\ln K - \theta - \mu_x}{\sigma_x}\right) = \frac{1}{2} \\ &\Leftrightarrow \theta = \ln K - \mu_x \\ &\Leftrightarrow \theta = \ln \frac{K}{S_0} - \left(r - \frac{1}{2}\sigma^2\right) T\end{aligned}$$

We deduce that:

$$\begin{aligned}\boldsymbol{\mathcal{P}} &= \mathbb{E}\left[\varphi\left(S\left(T\right)\right)\right] \\ &= \mathbb{E}\left[\varphi\left(S'\left(T\right)\right) \cdot \mathcal{L}\left(S'\left(T\right)\right)\right]\end{aligned}$$

where:

$$\begin{aligned}\mathcal{L}\left(x\right) &= \frac{\frac{1}{x\sigma_x}\phi\left(\frac{\ln x - \mu_x}{\sigma_x}\right)}{\frac{1}{x\sigma_z}\phi\left(\frac{\ln x - \mu_z}{\sigma_z}\right)} \\ &= \exp\left(\frac{\theta^2}{2\sigma_x^2} - \left(\frac{\ln x - \mu_x}{\sigma_x}\right) \cdot \frac{\theta}{\sigma_x}\right)\end{aligned}$$

and $S'\left(T\right)$ is the same geometric Brownian motion than $S\left(T\right)$, but with another initial value:

$$S'\left(0\right) = S\left(0\right) e^{\theta} = Ke^{-\left(r-\sigma^2/2\right)T}$$

Example 155 *We assume that* $S_0 = 100$, $K = 60$, $r = 5\%$, $\sigma = 20\%$ *and* $T = 2$. *If we consider the previous method, the IS process is simulated using the initial value* $S'\left(0\right) = Ke^{-\left(r-\sigma^2/2\right)T} = 56.506$, *whereas the value of* θ *is equal to* -0.5708. *In Figure 13.42, we report the density function of the estimators* $\hat{\boldsymbol{\mathcal{P}}}_{\text{MC}}$ *and* $\hat{\boldsymbol{\mathcal{P}}}_{\text{IS}}$ *when the number of simulations is equal to* 1 000. *For this example, the variance ratio is equal to* 1.77%, *meaning that the IS method has reduced the variance of the MC estimator by a factor greater than* 50. *If we use another IS scheme with* $S'\left(0\right) = 80$, *the reduction is less important, but remains significant*[43].

13.3.2.4 Other methods

We mention two other methods, which are less used in risk management than the previous methods, but may be very efficient for some financial problems. The first method is known as the conditional Monte Carlo method. Recall that $I = \mathbb{E}\left[Y\right]$ where $Y = \varphi\left(X_1, \ldots, X_n\right)$. Let Z be a random vector and $V = \mathbb{E}\left[Y \mid Z\right]$ be the conditional expectation of Y with respect to Z. It follows that:

$$\begin{aligned}\mathbb{E}\left[V\right] &= \int \mathbb{E}\left[Y \mid Z\right] f_z\left(z\right) \mathrm{d}z \\ &= \mathbb{E}\left[Y\right] \\ &= I\end{aligned}$$

[43] The variance ratio is equal to 13.59%.

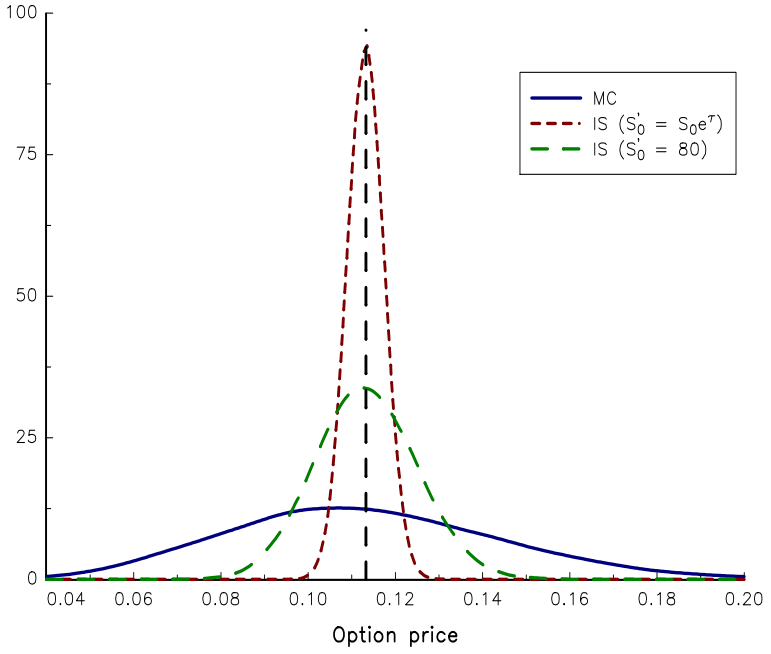

FIGURE 13.42: Density function of the estimators $\hat{\mathcal{P}}_{\text{MC}}$ and $\hat{\mathcal{P}}_{\text{IS}}$ $(n_S = 1\,000)$

where f_z is the probability density function of Z. Recall that:

$$\text{var}(Y) = \mathbb{E}[\text{var}(Y \mid Z)] + \text{var}(\mathbb{E}[Y \mid Z])$$

We deduce that:

$$\begin{aligned} \text{var}(V) &= \text{var}(\mathbb{E}[Y \mid Z]) \\ &= \text{var}(Y) - \mathbb{E}[\text{var}(Y \mid Z)] \\ &\leq \text{var}(Y) \end{aligned}$$

because $\text{var}(Y \mid Z) \geq 0$ implies that $\mathbb{E}[\text{var}(Y \mid Z)] \geq 0$. The idea of the conditional Monte Carlo method is then simulating V instead of Y in order to reduce the variance. For that, we have to find Z such that $\mathbb{E}[Y \mid Z]$ can easily be sampled. It can be the case with some stochastic volatility models.

Example 156 *Let $X = (X_1, X_2)$ be a standardized Gaussian random vector with correlation ρ. We want to calculate $p = \Pr\{X_1 \leq aX_2 + b\}$. We have:*

$$\begin{aligned} Y &= \varphi(X_1, X_2) \\ &= \mathbb{1}\{X_1 \leq aX_2 + b\} \end{aligned}$$

and:

$$\hat{p}_{\text{MC}} = \frac{1}{n_S} \sum_{s=1}^{n_S} \mathbb{1}\{X_{1,s} \leq aX_{2,s} + b\}$$

If we consider $Z = X_2$, we obtain:

$$\begin{aligned} V &= \mathbb{E}[Y \mid Z] \\ &= \mathbb{E}[\mathbb{1}\{X_1 \leq aX_2 + b\} \mid X_2 = x_2] \end{aligned}$$

Because we have $X_2 = \rho X_1 + \sqrt{1-\rho^2} X_3$ *where* $X_3 \sim \mathcal{N}(0,1)$ *is independent from* X_1*, we deduce that:*

$$\begin{aligned} V &= \mathbb{E}\left[\mathbb{1}\left\{X_1 \leq a\left(\rho X_1 + \sqrt{1-\rho^2} X_3\right) + b\right\} \mid X_3 = x_3\right] \\ &= \Phi\left(\frac{a\sqrt{1-\rho^2} x_3 + b}{1 - a\rho}\right) \end{aligned}$$

The conditional Monte Carlo (CMC) estimator is then equal to[44]*:*

$$\hat{p}_{\text{CMC}} = \frac{1}{n_S} \sum_{s=1}^{n_S} \Phi\left(\frac{a\sqrt{1-\rho^2} X_{3,s} + b}{1 - a\rho}\right)$$

where $X_{3,s} \sim \mathcal{N}(0,1)$*. In Table 13.6, we report the variance ratio between the CMC and MC estimators when* a *is equal to* 1*. We verify that the CMC estimator is particularly efficient when* ρ *is negative. For instance, the variance is divided by a factor of* 70 *when* ρ *is equal to* -90% *and* b *is equal to* 3.0*.*

TABLE 13.6: Variance ratio (in %) when $a = 1$

b	Correlation ρ (in %)								
	-90.0	-75.0	-50.0	-25.0	0.0	25.0	50.0	75.0	90.0
0.0	3.2	8.0	16.1	24.5	33.3	43.0	54.0	67.8	79.8
1.0	2.9	7.3	14.8	22.5	30.6	39.3	48.9	60.0	67.6
2.0	2.2	5.6	11.3	17.2	23.3	29.6	35.9	41.9	48.8
3.0	1.4	3.4	7.0	10.7	14.4	18.1	21.4	24.6	

The second method is the stratified sampling. Recall that $X \in \Omega$. Let $\{\Omega_j, j = 1, \ldots, m\}$ be a partition[45] of Ω. We have:

$$\begin{aligned} I &= \mathbb{E}[\varphi(X)] \\ &= \int_\Omega \varphi(x) f(x)\, \mathrm{d}x \\ &= \sum_{j=1}^m \int_{\Omega_j} \varphi(x) f(x)\, \mathrm{d}x \\ &= \sum_{j=1}^m \mathbb{E}[\mathbb{1}\{X \in \Omega_j\} \cdot \varphi(X)] \\ &= \sum_{j=1}^m \Pr\{X \in \Omega_j\} \cdot \mathbb{E}[\varphi(X) \mid X \in \Omega_j] \end{aligned}$$

We introduce the index random variable B:

$$B = j \Leftrightarrow X \in \Omega_j$$

[44] If $a\rho = 1$, we have:

$$\hat{p}_{\text{CMC}} = \frac{1}{n_S} \sum_{s=1}^{n_S} \mathbb{1}\left\{a\sqrt{1-\rho^2} X_{3,s} + b \geq 0\right\} = \hat{p}_{\text{MC}}$$

[45] This means that $\Omega_j \bigcap \Omega_k = \emptyset$ and $\bigcup_{j=1}^m \Omega_j = \Omega$.

We note $p(j) = \Pr\{B = j\} = \Pr\{X \in \Omega_j\}$ and $X(j)$ the random vector, whose probability distribution is the conditional law of $X \mid X \in \Omega_j$. It follows that:

$$\begin{aligned} I &= \sum_{j=1}^{m} p(j) \cdot \mathbb{E}[\varphi(X(j))] \\ &= \sum_{j=1}^{m} p(j) \cdot I(j) \end{aligned}$$

where $I(j) = \mathbb{E}[\varphi(X(j))] = \mathbb{E}[\varphi(X) \mid B = j]$. We define the stratified sampling estimator as follows:

$$\hat{I}_{\text{STR}} = \sum_{j=1}^{m} p(j) \cdot \hat{Y}(j)$$

where:

$$\hat{Y}(j) = \frac{1}{n_S(j)} \sum_{s=1}^{n_S(j)} Y_s(j) = \frac{1}{n_S(j)} \sum_{s=1}^{n_S(j)} \varphi(X_s(j))$$

Recall that the MC estimator is equal to:

$$\hat{I}_{\text{MC}} = \hat{Y}$$

where:

$$\hat{Y} = \frac{1}{n_S} \sum_{s=1}^{n_S} Y_s = \frac{1}{n_S} \sum_{s=1}^{n_S} \varphi(X_s)$$

The MC estimator can be viewed as a stratified sampling estimator with only one stratum: $\Omega_1 = \Omega$. On the contrary, the STR estimator depends on the number m of strata and the distribution of strata.

Like the MC estimator, it is easy to show that the stratified sampling estimator is unbiased[46]:

$$\mathbb{E}\left[\hat{I}_{\text{STR}}\right] = I$$

We introduce the following notations.

1. the conditional expectation $\mu(j)$ is defined as:

$$\mu(j) = \mathbb{E}[\varphi(X(j))] = \mathbb{E}[\varphi(X) \mid B = j]$$

2. the conditional variance $\sigma^2(j)$ is equal to:

$$\sigma^2(j) = \text{var}(\varphi(X(j))) = \text{var}(\varphi(X) \mid B = j)$$

Using the conditional independence of the random variables $X(j)$, it follows that:

$$\text{var}\left(\hat{I}_{\text{STR}}\right) = \sum_{j=1}^{m} \frac{p^2(j) \cdot \sigma^2(j)}{n_S(j)} \tag{13.11}$$

and:

$$\text{var}\left(\hat{I}_{\text{MC}}\right) = \frac{1}{n_S} \left(\sum_{j=1}^{m} p(j) \cdot \sigma^2(j) + \sum_{j=1}^{m} p(j) \cdot (\mu(j) - \bar{\mu})^2 \right) \tag{13.12}$$

[46] We assume that $n_S(j) \neq 0$.

where:

$$\bar{\mu} = \sum_{j=1}^{m} p(j) \cdot \mu(j)$$

Using Equations (13.11) and (13.12), it is not possible to compare directly the variance of the two estimators because the stratified sampling estimator depends on the allocation $(n_S(1), \ldots, n_S(m))$. Therefore, we can have $\text{var}\left(\hat{I}_{\text{STR}}\right) < \text{var}\left(\hat{I}_{\text{MC}}\right)$ or $\text{var}\left(\hat{I}_{\text{STR}}\right) > \text{var}\left(\hat{I}_{\text{MC}}\right)$. However, for many allocation schemes, the stratified sampling approach is an efficient method to reduce the variance of the MC estimator.

To illustrate the interest of the stratified sampling approach, we consider the proportional allocation:

$$n_S(j) = n_S \cdot p(j)$$

It follows that:

$$\text{var}\left(\hat{I}_{\text{STR}}\right) = \frac{1}{n_S} \sum_{j=1}^{m} p(j) \cdot \sigma^2(j)$$

and:

$$\begin{aligned}
\text{var}\left(\hat{I}_{\text{MC}}\right) &= \frac{1}{n_S} \sum_{j=1}^{m} p(j) \cdot \sigma^2(j) + \frac{1}{n_S} \sum_{j=1}^{m} p(j) \cdot (\mu(j) - \bar{\mu})^2 \\
&= \text{var}\left(\hat{I}_{\text{STR}}\right) + \frac{1}{n_S} \sum_{j=1}^{m} p(j) \cdot (\mu(j) - \bar{\mu})^2 \\
&\geq \text{var}\left(\hat{I}_{\text{STR}}\right)
\end{aligned}$$

Therefore, the stratified sampling estimator has a lower variance than the Monte Carlo estimator. In this case, we notice that:

$$\text{var}\left(\hat{I}_{\text{STR}}\right) = \frac{1}{n_S} \cdot \mathbb{E}\left[\text{var}\,\varphi(X) \mid B\right]$$

and:

$$\text{var}\left(\hat{I}_{\text{MC}}\right) = \frac{1}{n_S} \cdot \underbrace{\mathbb{E}\left[\text{var}\,\varphi(X) \mid B\right]}_{\text{intra-strata variance}} + \frac{1}{n_S} \cdot \underbrace{\text{var}\left(\mathbb{E}\left[\varphi(X) \mid B\right]\right)}_{\text{inter-strata variance}}$$

The stratified sampling approach with a proportional allocation consists of removing the inter-strata variance in order to keep only the intra-strata variance. This result gives some ideas about the optimal strata. Indeed, the variance reduction is high if the intra-strata variance is low.

We now consider that the strata are given. We write the allocation as follows:

$$n_S(j) = n_S \cdot q(j)$$

where the $q(j)$'s are arbitrary frequencies such that $\sum_{j=1}^{m} q(j) = 1$. To find the optimal allocation $q^\star$, we have to solve the following variance minimization problem:

$$q^\star = \arg\min \text{var}\left(\hat{I}_{\text{STR}}\right)$$

subject to the constraint $\sum_{j=1}^{m} q(j) = 1$. It follows that the Lagrange function is equal to:

$$\mathcal{L}(q; \lambda) = \frac{1}{n_S} \sum_{j=1}^{m} \frac{p^2(j) \cdot \sigma^2(j)}{q(j)} + \lambda \left(\sum_{j=1}^{m} q(j) - 1 \right)$$

We deduce that the optimal allocation is[47]:

$$q^{\star}(j) = \frac{p(j) \cdot \sigma(j)}{\sum_{j=1}^{m} p(j) \cdot \sigma(j)}$$

In this case, we obtain:

$$\begin{aligned} \text{var}\left(\hat{I}_{\text{STR}}\right) &= \frac{1}{n_S} \sum_{j=1}^{m} \frac{p^2(j) \cdot \sigma^2(j)}{q^{\star}(j)} \\ &= \frac{1}{n_S} \left(\sum_{j=1}^{m} p(j) \cdot \sigma(j) \right)^2 \end{aligned}$$

Example 157 *We have* $I = \int_0^1 \varphi(x)\,\mathrm{d}x = \mathbb{E}[\varphi(X)]$ *where* $X \sim \mathcal{U}_{[0,1]}$. *We consider the following cases for the function* $\varphi(x)$*:*

1. $\varphi(x) = x$
2. $\varphi(x) = x^2$
3. $\varphi(x) = (1 + \cos(\pi x))/2$

These three functions are reported in Figure 13.43. In Table 13.7, we give the exact value of I *and the variance of the estimators. For the MC estimator and the function* $\varphi(x) = x$*, we verify that:*

$$n_S \cdot \text{var}\left(\hat{I}_{\text{MC}}\right) = \text{var}\left(\mathcal{U}_{[0,1]}\right) = \frac{1}{12}$$

For the STR estimator, we consider fixed-space strata $X(j) \in \left[\frac{j-1}{m}, \frac{j}{m}\right]$ *implying that* $p(j) = 1/m$*. We can then simulate the conditional random variable* $X(j)$ *by using the following transformation:*

$$X(j) = \frac{j-1}{m} + \frac{U}{m}$$

where $U \sim \mathcal{U}_{[0,1]}$. *In Table 13.7, we report* $n_S \cdot \text{var}\left(\hat{I}_{\text{STR}}\right)$ *when* m *is equal to* 10. *We notice that the variance is approximately divided by* 100 *when we consider the proportional allocation* $q(j) = p(j)$. *To understand this result, we consider the function* $\varphi(x) = x$. *In this case, the variance of the stratum* j *is equal to*[48]*:*

$$\begin{aligned} \sigma^2(j) &= \mathbb{E}\left[X^2(j)\right] - \mathbb{E}^2\left[X(j)\right] \\ &= \int_{\frac{j-1}{m}}^{\frac{j}{m}} mx^2\,\mathrm{d}x - \left(\int_{\frac{j-1}{m}}^{\frac{j}{m}} mx\,\mathrm{d}x \right)^2 \\ &= \left[m \cdot \frac{x^3}{3} \right]_{\frac{j-1}{m}}^{\frac{j}{m}} - \left(\left[m \cdot \frac{x^2}{2} \right]_{\frac{j-1}{m}}^{\frac{j}{m}} \right)^2 \end{aligned}$$

[47] The first-order condition is:

$$\frac{\partial \mathcal{L}(q;\lambda)}{\partial q(j)} = -\frac{1}{n_S} \cdot \frac{p^2(j) \cdot \sigma^2(j)}{q^2(j)} + \lambda = 0$$

implying that the ratio $\dfrac{p(j) \cdot \sigma(j)}{q(j)}$ is constant.

[48] The density function of $X(j)$ is $f(x) = m$.

We deduce that:

$$\begin{aligned}\sigma^2(j) &= \frac{1}{3m^2}\left(j^3-(j-1)^3\right)-\frac{1}{4m^2}(2j-1)^2 \\ &= \frac{1}{12m^2}\end{aligned}$$

This implies that the variance of strata is equal to the variance of the uniform random variable divided by a factor of m^2*. These intra-strata variances are given in Figure 13.44. For the function* $\varphi(x) = x^2$*, the variance of strata increases with the index* j*. This is normal if we consider the graphic representation of the function* $\varphi(x)$ *in Figure 13.43. Indeed, the curvature of* $\varphi(x)$ *implies that there is more variance when* x *increases. We have also calculated the variance of the STR estimator when we use the optimal allocation* $q^\star$*, which is reported in Figure 13.45. In this case, we allocate a more important number of simulations for strata that present more variance. This is perfectly normal, because these strata are more difficult to simulate than strata with low variance. However, the gain of the optimal allocation is not very significant with respect to the proportional allocation.*

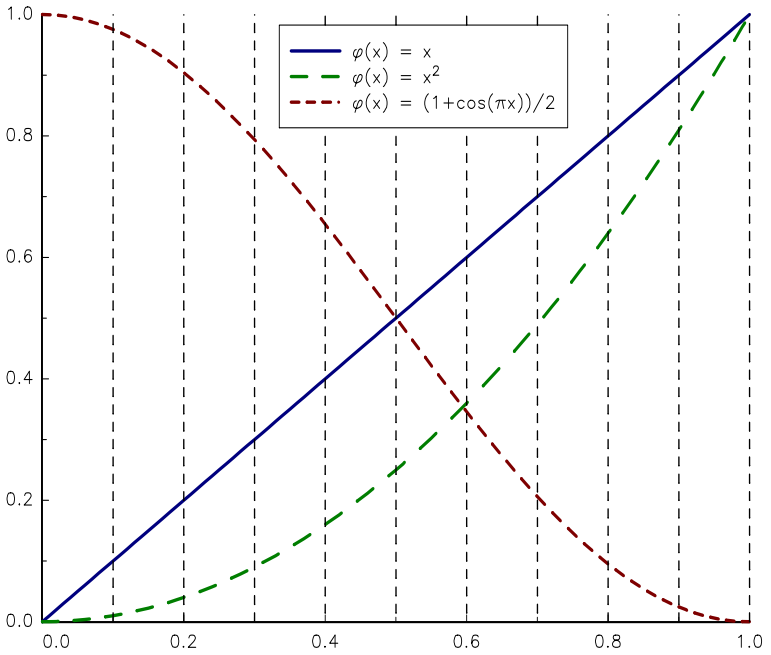

FIGURE 13.43: Function $\psi(x)$

In the case of the uniform distribution $\mathcal{U}_{[0,1]}$, we have used fixed-space strata $X(j) \in \left[\frac{j-1}{m}, \frac{j}{m}\right]$, implying that the probability $p(j)$ is equal for all the strata. This is the most popular method for defining strata. In the case of a general probability distribution $\mathbf{F}$, we define the conditional random variable $X(j)$ as follows:

$$X(j) = \mathbf{F}^{-1}\left(\frac{j-1}{m}+\frac{U}{m}\right) \tag{13.13}$$

TABLE 13.7: Comparison between MC and STR estimators

$\varphi(x)$		x	x^2	$(1+\cos(\pi x))/2$
I		0.50000	0.33333	0.50000
$n_S \text{ var}\left(\hat{I}_{\text{MC}}\right)$		0.08333	0.08890	0.12501
$n_S \text{ var}\left(\hat{I}_{\text{STR}}\right)$	$p(j)$	0.00083	0.00113	0.00105
$n_S \text{ var}\left(\hat{I}_{\text{STR}}\right)$	$q^\star(j)$	0.00083	0.00083	0.00084

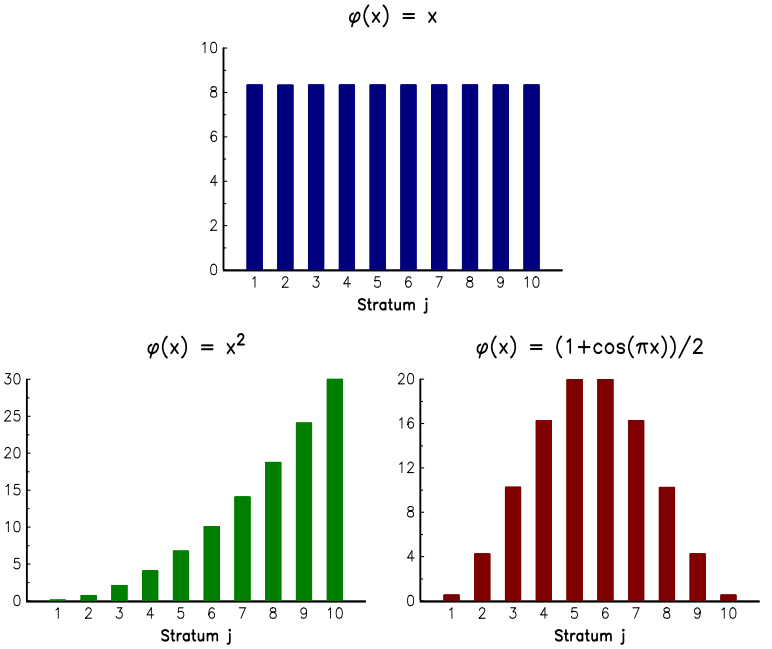

FIGURE 13.44: Intra-strata variance $\sigma^2(j)$ (in bps)

where U is a standard uniform random variate. We deduce that $X(j) \in \left[\mathbf{F}^{-1}\left(\frac{j-1}{m}\right), \mathbf{F}^{-1}\left(\frac{j}{m}\right)\right]$ and:

$$\begin{aligned} p(j) &= \Pr\left\{X \in \left[\mathbf{F}^{-1}\left(\frac{j-1}{m}\right), \mathbf{F}^{-1}\left(\frac{j}{m}\right)\right]\right\} \\ &= \mathbf{F}\left(\mathbf{F}^{-1}\left(\frac{j}{m}\right)\right) - \mathbf{F}\left(\mathbf{F}^{-1}\left(\frac{j-1}{m}\right)\right) \\ &= \frac{1}{m} \end{aligned}$$

In Figure 13.46, we have reported the strata defined by Equation (13.13) for different probability distribution when m is equal to 10.

The previous method consists in defining strata in order to obtain equal probabilities $p(j)$. It can be very different than the optimal method, whose objective is to define strata such that the intra-strata variances $\sigma(j)$ are close to zero. In order to illustrate the two approaches, we consider the pricing of an European call option in the Black-Scholes model. Recall that the price of the call option is equal to:

$$\mathcal{C} = e^{-rT}\mathbb{E}\left[\max\left(0, S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)T+\sigma\sqrt{T}X} - K\right)\right]$$

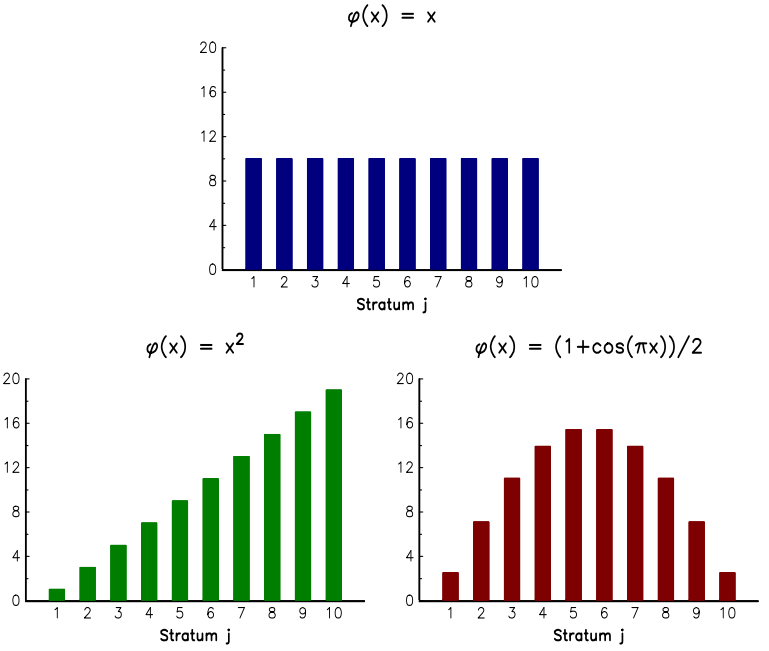

FIGURE 13.45: Optimal allocation $q^\star(j)$ (in %)

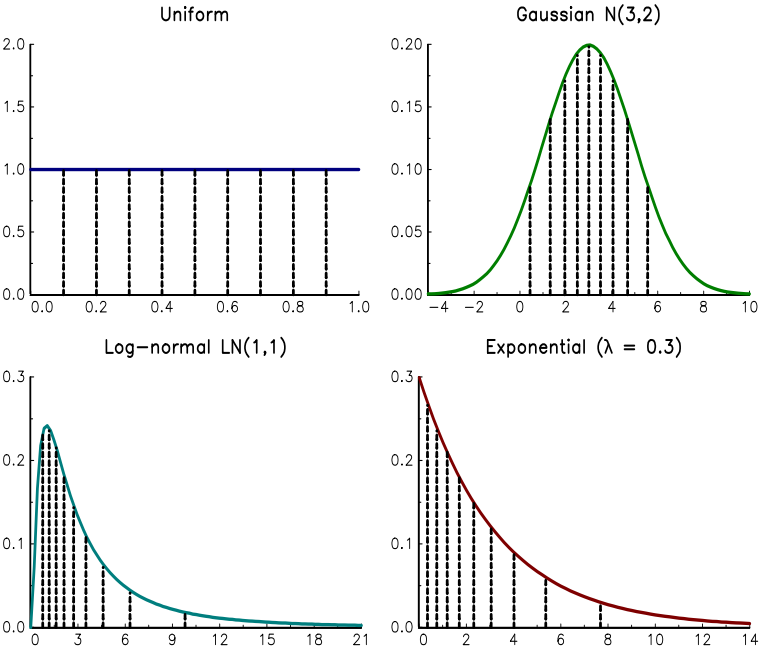

FIGURE 13.46: Strata for different random variables

where $X \sim \mathcal{N}(0,1)$. Let us apply the stratification method with strata defined by Equation (13.13). We have:

$$X(j) = \Phi^{-1}\left(\frac{j-1}{m} + \frac{U}{m}\right)$$

where $U \sim \mathcal{U}_{[0,1]}$. We deduce that:

$$\hat{I}^{(1)}_{\text{STR}}(m) = \frac{e^{-rT}}{m}\sum_{j=1}^{m}\frac{1}{n_S(j)}\sum_{s=1}^{n_s(j)}\max\left(0, S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)T+\sigma\sqrt{T}X_s(j)} - K\right)$$

However, we notice that $\max(0, S(T) - K)$ is equal to zero if:

$$\begin{aligned} & S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)T+\sigma\sqrt{T}X} - K \leq 0 \\ \Leftrightarrow \quad & X \leq x_1 = \frac{1}{\sigma\sqrt{T}}\left(\ln\left(\frac{K}{S_0}\right) - rT\right) + \frac{1}{2}\sigma\sqrt{T} \end{aligned}$$

It is then natural to define the first stratum by $]-\infty, x_1]$. Indeed, we have $\varphi(X(1)) = 0$, implying that the intra-strata variance $\sigma(1)$ is equal to zero. For the other strata, we can use the previous approach. For $j \geq 2$, we have:

$$p(j) = \Pr\{X \leq x_j\} - \Pr\{X \leq x_{j-1}\} = \frac{1-p(1)}{m-1}$$

We deduce that:

$$\Pr\{X \leq x_j\} = p(1) + \frac{j-1}{m-1}\cdot(1-p(1))$$

and:

$$x_j = \Phi^{-1}\left(p(1) + \frac{j-1}{m-1}\cdot(1-p(1))\right)$$

The j^{th} stratum is then defined by $X \in [x_{j-1}, x_j]$ with $x_0 = -\infty$, $x_m = +\infty$ and:

$$p(j) = \begin{cases} p(1) & \text{if} \quad j = 1 \\ \dfrac{1-p(1)}{m-1} & \text{if} \quad j \geq 2 \end{cases}$$

To simulate the conditional random variable $X(j)$ for $j \geq 2$, we use the following scheme:

$$X(j) = \Phi^{-1}\left(\Phi(x_{j-1}) + (\Phi(x_j) - \Phi(x_{j-1}))\cdot U\right)$$

where $U \sim \mathcal{U}_{[0,1]}$. We deduce that:

$$\hat{I}^{(2)}_{\text{STR}}(m) = \left(\frac{1-p(1)}{m-1}\right)e^{-rT}\sum_{j=2}^{m}\frac{1}{n_S(j)}\sum_{s=1}^{n_s(j)}\left(S_0 e^{\left(r-\frac{1}{2}\sigma^2\right)T+\sigma\sqrt{T}X_s(j)} - K\right)$$

In the case of proportional allocation $n_S(j) = n_S \cdot p(j)$, we notice that the total number of simulations is reduced and equal to $(1-p(1))\cdot n_S$ because we don't have to simulate the first stratum.

We consider an European call option with the following parameters: $S_0 = 100$, $K = 105$, $\sigma = 20\%$, $\tau = 1$ and $r = 5\%$. In Figure 13.47, we have reported the variance of the two stratified estimators for different values of m when the number of simulations n_S is equal to 10 000. In the case $m = 1$, we obtain the traditional MC estimator and we have:

$$\text{var}\left(\hat{I}^{(2)}_{\text{STR}}(1)\right) = \text{var}\left(\hat{I}^{(1)}_{\text{STR}}(1)\right) = \text{var}\left(\hat{I}_{\text{MC}}\right)$$

When $m > 1$, we obtain:

$$\text{var}\left(\hat{I}_{\text{STR}}^{(2)}(m)\right) < \text{var}\left(\hat{I}_{\text{STR}}^{(1)}(m)\right) < \text{var}\left(\hat{I}_{\text{MC}}\right)$$

The second stratified estimator is then more efficient than the first stratified estimator, because the design of the first stratum is optimal[49].

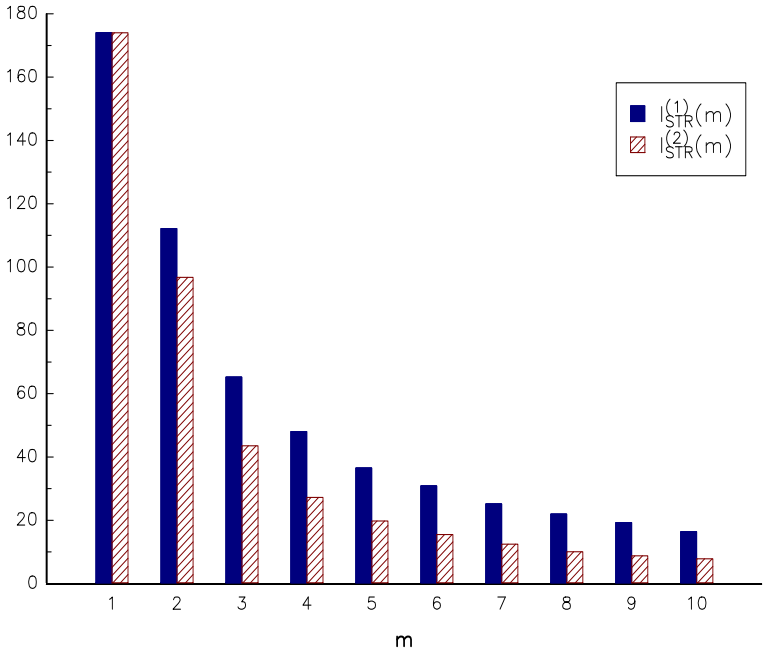

FIGURE 13.47: Variance of the two estimators $\hat{I}_{\text{STR}}^{(1)}(m)$ and $\hat{I}_{\text{STR}}^{(2)}(m)$ for different values of m

13.3.3 MCMC methods

Let us consider a Markov chain with transition density $p\left(x^{(t+1)} \mid x^{(t)}\right) = \Pr\left\{X^{(t+1)} = x^{(t+1)} \mid X^{(t)} = x^{(t)}\right\}$. We also assume that it has a stationary distribution $\pi(x)$. In this case, we can show that the Markov chain satisfies the detailed balance equation[50]:

$$p(y \mid x) \cdot \pi(x) = p(x \mid y) \cdot \pi(y) \tag{13.14}$$

It follows that:

$$\int p(x \mid y)\, \pi(y)\, \mathrm{d}y = \int p(y \mid x)\, \pi(x)\, \mathrm{d}y = \pi(x) \tag{13.15}$$

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for simulating a sample from a probability density function $f(x)$. The underlying idea is to simulate a Markov chain, whose limiting pdf is the desired pdf $f(x)$. It is then equivalent to find the

[49] We have $x_1 = 0.093951$ and $p_1 = 53.74\%$.

[50] In the discrete case, we have:

$$p_{i,j} \cdot \pi_i = p_{j,i} \cdot \pi_j$$

for all the states (i, j) of the Markov chain.

transition kernel $\kappa\left(x^{(t+1)} \mid x^{(t)}\right)$ such that the detailed balance property is satisfied:

$$\kappa(y \mid x) \cdot f(x) = \kappa(x \mid y) \cdot f(y) \tag{13.16}$$

In this case, MCMC methods then generate a sample such that:

$$\lim_{t \to \infty} \Pr\left\{X^{(t)} = x\right\} = f(x)$$

Because the solution of Equation (13.16) is not unique, there are different MCMC methods that will differ by the specification of the transition kernel $\kappa(y \mid x)$.

13.3.3.1 Gibbs sampling

According to Casella and George (1992), the Gibbs sampler has been formulated by Geman and Geman (1984), who studied image-processing models, and popularized in statistics by Gelfand and Smith (1990). Let $f(x_1, \ldots, x_n)$ be the target probability density function and $f(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ the conditional density of the i^{th} component of the random vector $X = (X_1, \ldots, X_n)$. At iteration t, the Gibbs sampler (GS) is given by the following steps:

- we draw $x_1^{(t)} \sim f\left(x_1 \mid x_2^{(t-1)}, \ldots, x_n^{(t-1)}\right)$;
- we draw $x_2^{(t)} \sim f\left(x_2 \mid x_1^{(t)}, x_3^{(t-1)}, \ldots, x_n^{(t-1)}\right)$;
- we draw $x_3^{(t)} \sim f\left(x_3 \mid x_1^{(t)}, x_2^{(t)}, x_4^{(t-1)}, \ldots, x_n^{(t-1)}\right)$;
- we draw

$$x_i^{(t)} \sim f\left(x_i \mid \underbrace{x_1^{(t)}, \ldots, x_{i-1}^{(t)}}_{\text{iteration } t}, \underbrace{x_{i+1}^{(t-1)}, \ldots, x_n^{(t-1)}}_{\text{iteration } t-1}\right)$$

  for $i = 4, \ldots, n-1$;
- we draw $x_n^{(t)} \sim f\left(x_n \mid x_1^{(t)}, \ldots, x_{n-1}^{(t)}\right)$;

The algorithm is initialized with $\left(x_1^{(0)}, \ldots, x_n^{(0)}\right)$. After t iterations, we obtain the following Gibbs sequence:

$$\left\{\left(x_1^{(1)}, \ldots, x_n^{(1)}\right), \ldots, \left(x_1^{(t)}, \ldots, x_n^{(t)}\right)\right\} \tag{13.17}$$

Under some conditions and if t is sufficiently large, $\left(x_1^{(t)}, \ldots, x_n^{(t)}\right)$ is a random sample of the joint distribution $f(x_1, \ldots, x_n)$. To obtain n_S simulations of the density $f(x_1, \ldots, x_n)$, Gelfand and Smith (1990) suggested then to generate n_S Gibbs sequences and to take the final value $\left(x_{1,s}^{(t)}, \ldots, x_{n,s}^{(t)}\right)$ from each sequence s. The Monte Carlo estimator of $\mathbb{E}\left[\varphi(X_1, \ldots, X_n)\right]$ is then equal to:

$$\hat{I}_{n_S} = \frac{1}{n_S} \sum_{s=1}^{n_S} \varphi\left(X_{1,s}^{(t)}, \ldots, X_{n,s}^{(t)}\right) \tag{13.18}$$

However, if the Markov chain has reached his stationary state at time n_b, this implies that the Gibbs sequence:

$$\left\{\left(x_1^{(n_b+1)}, \ldots, x_n^{(n_b+1)}\right), \ldots, \left(x_1^{(n_b+n_S)}, \ldots, x_n^{(n_b+n_S)}\right)\right\} \tag{13.19}$$

is also a Gibbs sample of the density $f(x_1, \ldots, x_n)$. We can then formulate another MC estimator[51]:

$$\hat{I}_{n_S} = \frac{1}{n_S} \sum_{t=1}^{n_S} \varphi\left(X_1^{(n_b+t)}, \ldots, X_n^{(n_b+t)}\right) \tag{13.20}$$

However, contrary to the Gelfand-Smith approach, the random vectors of this estimator are correlated. Therefore, the variance of this estimator is larger than in the independent case.

Let us consider the two-dimensional case. We note (X, Y) the random vector and $f(x, y)$ the targeted distribution. At time t, we have $X^{(t)} = x$ and $Y^{(t)} = y$, and we assume that $X^{(t+1)} = x'$ and $Y^{(t+1)} = y'$. If $(x, y) \sim f$, the density $g(x', y')$ of the Gibbs sample is equal to[52]:

$$\begin{aligned} g(x', y') &= \int f(x, y)\, f(x' \mid y)\, f(y' \mid x')\, \mathrm{d}x\, \mathrm{d}y \\ &= \int f(x, y)\, \frac{f(x', y)}{f_y(y)}\, \frac{f(y', x')}{f_x(x')}\, \mathrm{d}x\, \mathrm{d}y \end{aligned}$$

where f_x and f_y are the marginal densities of the joint distribution $f(x, y)$. It follows that:

$$\begin{aligned} g(x', y') &= \int \frac{f(x, y)}{f_y(y)}\, \frac{f(x', y)}{f_x(x')}\, f(y', x')\, \mathrm{d}x\, \mathrm{d}y \\ &= \int f(x \mid y)\, f(y \mid x')\, f(y', x')\, \mathrm{d}x\, \mathrm{d}y \\ &= f(y', x') \int f(x \mid y)\, f(y \mid x')\, \mathrm{d}x\, \mathrm{d}y \end{aligned}$$

Because the events $\{X \mid Y = y\}$ and $\{Y \mid X = x'\}$ are independent, we obtain:

$$\begin{aligned} g(x', y') &= f(y', x') \int f(x \mid y)\, \mathrm{d}x \int f(y \mid x')\, \mathrm{d}y \\ &= f(y', x') \end{aligned}$$

We deduce that $(x', y') \sim f$. If the Gibbs sampler reaches a stationary regime, it then converges to the targeted distribution $f(x, y)$.

Remark 169 *In the two dimensional case, we notice that the proposal kernel is equal to:*

$$\kappa(x', y' \mid x, y) \propto f(x' \mid y) \cdot f(y' \mid x')$$

In the general case, we have:

$$\kappa(y \mid x) \propto \prod_{i=1}^{n} f(y_i \mid y_1, \ldots, y_{i-1}, x_{i+1}, \ldots, x_n)$$

Example 158 *Casella and George (1992) consider the following joint distribution of X and Y:*

$$f(x, y) \propto \binom{n}{x} y^{x+\alpha-1} (1-y)^{n-x+\beta-1}$$

[51] This approach requires a burn-in period, meaning that an initial number of samples is discarded. However, it is not always obvious to know when the Markov chain has converged.

[52] We have the following sequences $(x, y) \to (x', y)$ where $x' \sim f(X \mid y)$ and $(x', y) \to (x', y')$ where $y' \sim f(Y \mid x')$.

where $x \in \{0, 1, \ldots, n\}$ *and* $y \in [0, 1]$. *It follows that:*

$$\begin{aligned} f(x \mid y) &\propto y^{\alpha-1}(1-y)^{\beta-1} \cdot \binom{n}{x} y^x (1-y)^{n-x} \\ &\propto \binom{n}{x} y^x (1-y)^{n-x} \\ &\sim \mathcal{B}(n, y) \end{aligned}$$

and:

$$\begin{aligned} f(y \mid x) &\propto \binom{n}{x} \cdot y^{x+\alpha-1}(1-y)^{n-x+\beta-1} \\ &\propto y^{x+\alpha-1}(1-y)^{n-x+\beta-1} \\ &\sim \mathcal{B}(x+\alpha, n-x+\beta) \end{aligned}$$

Therefore, $\{X \mid Y = y\}$ *is a Bernoulli random variable* $\mathcal{B}(n, p)$ *with* $p = y$ *and* $\{Y \mid X = x\}$ *is a beta random variable* $\mathcal{B}(\alpha', \beta')$ *with* $\alpha' = x + \alpha$ *and* $\beta' = n - x + \beta$. *In Figure 13.48, we have reported the Gibbs sequence of* 1 000 *iterations* $\left(x^{(t)}, y^{(t)}\right)$ *for the parameters* $n = 5$, $\alpha = 2$ *and* $\beta = 4$. *The initial values are* $x^{(0)} = 5$ *and* $y^{(0)} = 1/2$. *We assume that the burn-in-period corresponds to the initial 200 iterations. We can then calculate* $I = \mathbb{E}[X \cdot Y]$ *by Monte Carlo with the next* 800 *iterations. We obtain* $\hat{I} = 0.71$. *We can also show that the variance of this MCMC estimator is three times larger than the variance of the traditional MC estimator. This is due to the high autocorrelation between the samples*[53].

13.3.3.2 Metropolis-Hastings algorithm

Like the Gibbs sampler, the Metropolis-Hastings algorithm considers a multidimensional probability density function $f(x) = f(x_1, \ldots, x_n)$. Let $q(y \mid x) = q(y_1, \ldots, y_n \mid x_1, \ldots, x_n)$ be the Markov transition density or the proposal density. The Metropolis-Hastings (MH) algorithm consists in the following steps:

1. given the state $x^{(t)}$, we generate $y \sim q\left(Y \mid x^{(t)}\right)$ from the Markov transition density;
2. we generate a uniform random number $u \sim \mathcal{U}_{[0,1]}$;
3. we calculate the density ratio $r\left(x^{(t)}, y\right)$ defined by:

$$r(x, y) = \frac{q(x \mid y) \cdot f(y)}{q(y \mid x) \cdot f(x)}$$

   and $\alpha\left(x^{(t)}, y\right) = \min\left(r\left(x^{(t)}, y\right), 1\right)$;

4. we set:

$$x^{(t+1)} = \begin{cases} y & \text{if } u \leq \alpha\left(x^{(t)}, y\right) \\ x^{(t)} & \text{otherwise} \end{cases}$$

The Metropolis-Hastings algorithm can be viewed as an acceptance-rejection algorithm, when the samples are correlated due to the Markov chain (Hastings, 1970).

[53] We have:

$$\rho\left\langle X^{(t)} \cdot Y^{(t)}, X^{(t-1)} \cdot Y^{(t-1)} \right\rangle = 52\%$$

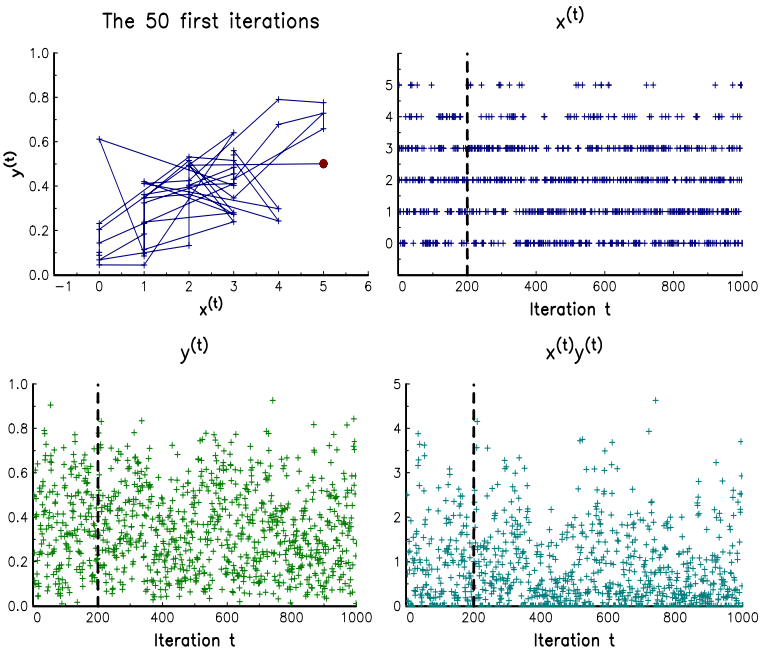

FIGURE 13.48: Illustration of the Gibbs sampler

The underlying idea of the MH algorithm is to build a kernel density $\kappa(y \mid x)$ such that the Markov chain converges to the targeted distribution $f(x)$. In this case, we must verify that:

$$\kappa(y \mid x) \cdot f(x) = \kappa(x \mid y) \cdot f(y)$$

It would be a pure coincidence that the kernel density $\kappa(y \mid x)$ is equal to the proposal density $q(y \mid x)$. Suppose that:

$$q(y \mid x) \cdot f(x) > q(x \mid y) \cdot f(y)$$

Chib and Greenberg (1995) explain that "*the process moves from x to y too often and from y to x too rarely*". To reduce the number of moves from x to y, we can introduce the probability $\alpha(x, y) < 1$ such that $\alpha(y, x) = 1$ and:

$$\begin{aligned} q(y \mid x) \cdot \alpha(x, y) \cdot f(x) &= q(x \mid y) \cdot \alpha(y, x) \cdot f(y) \\ &= q(x \mid y) \cdot f(y) \end{aligned}$$

where $\alpha(y, x) = 1$. We deduce that:

$$\alpha(x, y) = r(x, y) = \frac{q(x \mid y) \cdot f(y)}{q(y \mid x) \cdot f(x)}$$

If $q(y \mid x) \cdot f(x) < q(x \mid y) \cdot f(y)$, we have $q(y \mid x) \cdot f(x) = q(x \mid y) \cdot \alpha(y, x) \cdot f(y)$. Because the Markov chain must be reversible, we finally obtain that:

$$\alpha(x, y) = \min\left(\frac{q(x \mid y) \cdot f(y)}{q(y \mid x) \cdot f(x)}, 1\right)$$

From the previous analysis, we deduce that the kernel density is $\kappa(y \mid x) = q(y \mid x)\,\alpha(x, y)$. However, this result does not take into account that the Markov chain can remain at x.

Therefore, Chib and Greenberg (1995) show that the kernel density of the MH algorithm has the following form:

$$\begin{aligned}\kappa(y \mid x) &= q(y \mid x)\,\alpha(x,y) + \\ &\quad \left(1 - \int q(y \mid x)\,\alpha(x,y)\,\mathrm{d}y\right)\delta_x(y)\end{aligned}$$

where $\delta_x(y)$ is the Dirac delta function.

Remark 170 *The previous analysis shows that $\alpha(x,y)$ is the probability to move from x to y. $\alpha(x,y)$ is then the acceptance ratio of the MH algorithm. Contrary to the rejection sampling algorithm, the efficiency of the MH algorithm is not necessarily obtained when $\alpha(x,y)$ is equal to 1. Indeed, there are two sources of correlation between $x^{(t)}$ and $x^{(t+1)}$: (1) the correlation $\rho\left(x^{(t)},y\right)$ between $x^{(t)}$ and y, and (2) the correlation $\rho\left(x^{(t)},x^{(t)}\right)$ because y is rejected. Therefore, we face a trade-off between reducing the correlation $\rho\left(x^{(t)},y\right)$ and increasing the acceptance ratio $\alpha\left(x^{(t)},y\right)$. Suppose for instance that we use a proposal distribution with small variance, the correlation $\rho\left(x^{(t)},y\right)$ is high, but the acceptance ratio $\alpha\left(x^{(t)},y\right)$ is high. On the contrary, if we use a proposal distribution with small variance, the correlation $\rho\left(x^{(t)},y\right)$ is low, but the acceptance ratio $\alpha\left(x^{(t)},y\right)$ is also low. Therefore, it is extremely difficult to find a proposal distribution such that the correlation $\rho\left(x^{(t)},y\right)$ is low and the acceptance ratio $\alpha\left(x^{(t)},y\right)$ is high.*

In the original Metropolis algorithm (Metropolis *et al.*, 1953), the authors assumed that the proposal distribution is symmetric: $q(y \mid x) = q(x \mid y)$. In this case, the acceptance ratio is equal to:

$$\alpha(x,y) = \min\left(\frac{f(y)}{f(x)}, 1\right)$$

An example of such algorithm is the random walk sampler:

$$Y = x^{(t)} + Z$$

where the random vector Z follows a symmetric distribution. Another special case of the Metropolis-Hastings algorithm is the independence sampler: $q(y \mid x) = q(y)$. The proposal distribution does not depend on x and the acceptance ratio becomes:

$$\alpha(x,y) = \min\left(\frac{q(x)\cdot f(y)}{q(y)\cdot f(x)}, 1\right)$$

The MH algorithm is then very similar to the rejection sampling method, except that it produces correlated samples. We also notice that the Gibbs sampler is a special case of the MH algorithm where[54]:

$$q(y \mid x) \sim f(x_i \mid x_{-i})$$

Example 159 *We consider the simulation of the bivariate Gaussian random vector $X = (X_1, X_2) \sim \mathcal{N}(\mu, \Sigma)$ with the Metropolis-Hastings algorithm and a symmetric proposal distribution. It follows that:*

$$\alpha(x,y) = \min\left(\frac{\exp\left(-\frac{1}{2}(y-\mu)^\top \Sigma^{-1}(y-\mu)\right)}{\exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)}, 1\right)$$

[54] At each iteration, we have $y_i \sim f(x_i \mid x_{-i})$, $y_j = x_j$ if $j \neq i$, and $\alpha(x,y) = 1$.

The parameters are $\mu_1 = 1$, $\mu_2 = -1$, $\sigma_1 = 2$, $\sigma_2 = 1$ *and* $\rho = 99\%$. *We use the random walk sampler* $Y_i = x_i^{(t)} + Z_i$ *for* $i = 1, 2$. *The random vector* (Z_1, Z_2) *is generated using the following four proposal distributions:*

(a) $Z_1 \sim \mathcal{N}(0, 1)$ *and* $Z_2 \sim \mathcal{N}(0, 1)$;

(b) $Z_1 \sim \mathcal{N}(0, 0.1)$ *and* $Z_2 \sim \mathcal{N}(0, 0.1)$;

(c) $Z_1 \sim \mathcal{U}_{[-2,2]}$ *and* $Z_2 \sim \mathcal{U}_{[-2,2]}$;

(d) $Z_1 \sim \mathcal{U}_{[-0.2,0.2]}$ *and* $Z_2 \sim \mathcal{U}_{[-0.2,0.2]}$.

In Figure 13.49, we have reported the simulated samples of the first 2 000 *iterations for the four cases when we use a burn-in-period of* 500 *iterations. The sampler is initialized with* $x_1^{(0)} = x_2^{(0)} = 0$. *The acceptance ratio is respectively equal to* 15% *(a),* 43% *(b),* 10% *(c) and* 72% *(d). The acceptance ratio is the highest for the case (d). However, we notice that this proposal distribution is slow to explore the entire space. To obtain a sample such that* $x_1^{(t)} > 3$ *and* $x_2^{(t)} > 0$, *we need more iterations* ($n_S > 5000$). *On the contrary, proposal distributions (a) and (c) have a high variance. The exploration of the probability space is then faster, but the acceptance ratio is also lower. This example illustrates the trade-off between the autocorrelation and the acceptance ratio.*

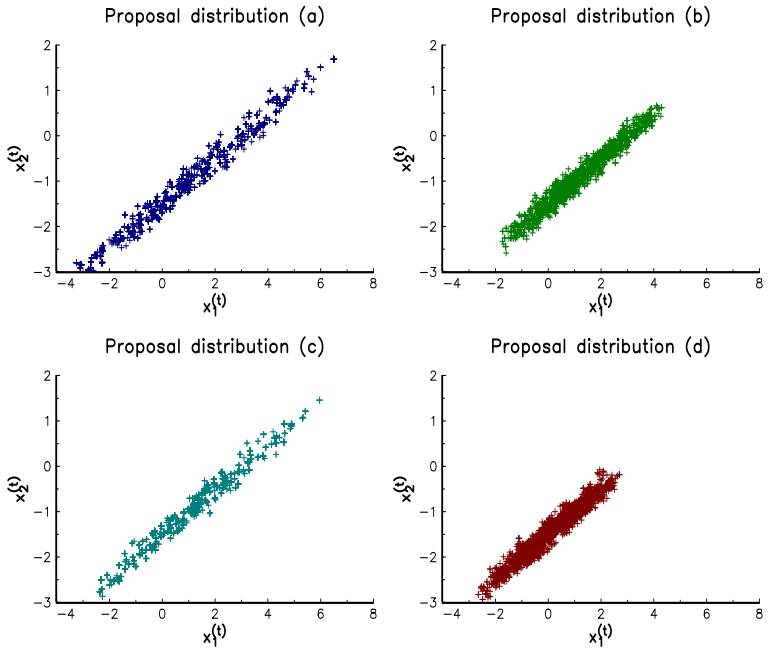

FIGURE 13.49: Illustration of the random walk sampler

The previous example is purely illustrative, because we don't need to use MCMC for simulating Gaussian random vectors. Let us now consider the following bivariate probability density functions:

(a) the pdf is a perturbation of the Gaussian density function:

$$f(x_1, x_2) \propto \exp\left(-x_1^2 - x_2^2 - x_1\right)$$

(b) the pdf is a mixture of two Gaussian density functions:

$$f(x_1, x_2) \propto \exp\left(-x_1^2 - x_2^2\right) + \exp\left(-x_1^2 - x_2^2 - 1.8 \cdot x_1 \cdot x_2\right)$$

(c) the pdf is a complex function of $\Phi(x_1)$ and $\Phi(x_2)$:

$$f(x_1, x_2) \propto \exp\left(-0.1 \cdot \Phi(x_1) \cdot \Phi(x_2) \cdot x_2^2\right)$$

(d) we consider the pdf of the Clayton copula with two exponential marginal distributions[55]:

$$f(x_1, x_2) = (1+\theta)\lambda_1\lambda_2 e^{-\lambda_1 x_1 - \lambda_2 x_2}\left(u_1^{-\theta} + u_2^{-\theta} - 1\right)^{-1/\theta - 2}(u_1 u_2)^{-1-\theta}$$

where $u_1 = 1 - e^{-\lambda_1 x_1}$ and $u_2 = 1 - e^{-\lambda_2 x_2}$.

We notice that we don't know the normalization constant for the first three pdfs. The third pdf is very complex and needs a very accurate algorithm for computing the Gaussian cdf. The fourth case is a copula model. We use the bivariate Gaussian probability distribution with a correlation equal to 50% for the proposal distribution. A sample of 1500 iterations simulated with the random walk sampler is given in Figure 13.50. The fourth panels have been obtained with the same random numbers of U, Z_1 and Z_2.

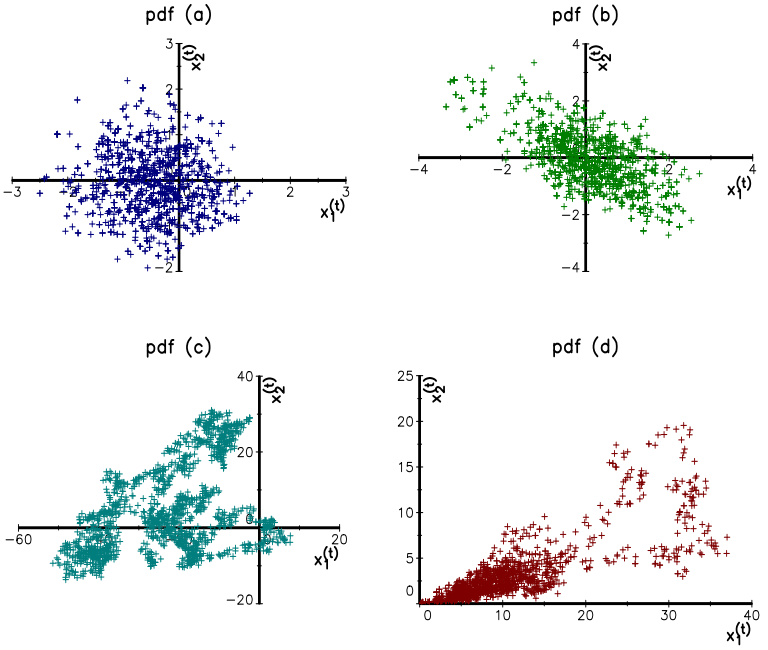

FIGURE 13.50: Simulating bivariate probability distributions with the MH algorithm

Remark 171 *For high-dimensional MCMC problems, the proposal distribution is generally the Gaussian distribution* $\mathcal{N}(\mu, \Sigma)$ *or the multivariate Student's t distribution* $t_n(\Sigma, \nu)$.

[55] We use the parameters $\theta = 2.5$, $\lambda_1 = 1\%$ and $\lambda_2 = 5\%$.

13.3.3.3 Sequential Monte Carlo methods and particle filters

Sequential Monte Carlo methods (or SMC) and particle filters (PF) are used when we consider non-linear/non-Gaussian state space models (or hidden Markov models). In these models, the state vector $X^{(t)}$ is characterized by the transition density:

$$X^{(t+1)} = x' \mid X^{(t)} = x \sim f\left(x' \mid x\right)$$

We assume that the state vector $X^{(t)}$ is not directly observed. The process that is observed is denoted by $Y^{(t)}$ and is characterized by the measurement density:

$$Y^{(t)} = y \mid X^{(t)} = x \sim f\left(y \mid x\right)$$

Let $x^{(0:T)} = \left(x^{(0)}, \ldots, x^{(T)}\right)$ be the exhaustive statistic of the system. By the Markov property, we have:

$$f\left(x^{(0:T)}\right) = f\left(x^{(0)}\right) \prod_{t=1}^{T} f\left(x^{(t)} \mid \mathcal{I}_t\right)$$

where $\mathcal{I}_t$ represents all the information available at time t (including $y^{(1)}, \ldots, y^{(t)}$). To calculate $f\left(x^{(t)} \mid \mathcal{I}_t\right)$, we have:

$$f\left(x^{(t)} \mid \mathcal{I}_t\right) \propto f\left(y^{(t)} \mid x^{(t)}\right) f\left(x^{(t)} \mid \mathcal{I}_{t-1}\right) \tag{13.21}$$

where $f\left(x^{(t)} \mid \mathcal{I}_{t-1}\right)$ is the prior density of $X^{(t)}$ and $f\left(y^{(t)} \mid x^{(t)}\right)$ is the log-likelihood function of the observed variable $Y^{(t)}$. This equation is known as the Bayes update step. We recall that the prior density of the state variable $X^{(t)}$ is given by the Chapman-Kolmogorov equation:

$$f\left(x^{(t)} \mid \mathcal{I}_{t-1}\right) = \int f\left(x^{(t)} \mid x^{(t-1)}\right) f\left(x^{(t-1)} \mid \mathcal{I}_{t-1}\right) \mathrm{d}x^{(t-1)} \tag{13.22}$$

This equation is also known as the Bayes prediction step. It gives an estimate of the probability density function of $X^{(t)}$ given all the information until $t-1$. A Bayesian filter corresponds to the system of the two recursive equations (13.21) and (13.22). In order to initialize the recurrence algorithm, we assume that the density function of the initial state vector $f\left(x^{(0)}\right)$ is known.

The underlying idea of SMC methods is then to estimate the density functions $f\left(x^{(t)} \mid \mathcal{I}_{t-1}\right)$ and $f\left(x^{(t)} \mid \mathcal{I}_t\right)$. Given these estimates, we may also compute the best estimates $\hat{x}^{(t)} \mid \mathcal{I}_{t-1}$ and $\hat{x}^{(t)}$, which are given by:

$$\hat{x}^{(t)} \mid \mathcal{I}_{t-1} = \mathbb{E}\left[X^{(t)} \mid \mathcal{I}_{t-1}\right] = \int x^{(t)} f\left(x^{(t)} \mid \mathcal{I}_{t-1}\right) \mathrm{d}x^{(t)}$$

and:

$$\hat{x}^{(t)} = \mathbb{E}\left[X^{(t)} \mid \mathcal{I}_t\right] = \int x^{(t)} f\left(x^{(t)} \mid \mathcal{I}_t\right) \mathrm{d}x^{(t)}$$

For that, we estimate the density $f\left(x^{(t)} \mid \mathcal{I}_t\right)$ by the Monte Carlo method. At time $t-1$, we assume that $f\left(x^{(t-1)} \mid \mathcal{I}_{t-1}\right)$ is approximated by a sample $\left\{x_1^{(t-1)}, \ldots, x_{n_S}^{(t-1)}\right\}$ and a vector of associated weights $\left\{w_1^{(t-1)}, \ldots, w_{n_S}^{(t-1)}\right\}$, where n_S is the number of simulated particles. We deduce that:

$$f\left(x^{(t)} \mid \mathcal{I}_t\right) \propto f\left(y^{(t)} \mid x^{(t)}\right) \sum_{s=1}^{n_S} w_s^{(t-1)} f\left(x^{(t)} \mid x_s^{(t-1)}\right)$$

The problem consists then in estimating the states $\left\{x_1^{(t)}, \ldots, x_{n_S}^{(t)}\right\}$ and the corresponding weights $\left\{w_1^{(t)}, \ldots, w_{n_S}^{(t)}\right\}$.

Computation of weights Following Arulampalam *et al.* (2002), we apply the method of importance sampling to the joint distribution of $x^{(0:t)} = \left(x^{(0)}, \ldots, x^{(t)}\right)$:

$$\begin{aligned} f\left(x^{(0:t)} \mid y^{(t)}, \mathcal{I}_{t-1}\right) &= f\left(x^{(t)} \mid y^{(t)}, x^{(0:t-1)}, \mathcal{I}_{t-1}\right) f\left(x^{(0:t-1)} \mid \mathcal{I}_{t-1}\right) \\ &= \frac{f\left(y^{(t)} \mid x^{(t)}\right) f\left(x^{(t)} \mid x^{(0:t-1)}\right)}{f\left(y^{(t)} \mid \mathcal{I}_{t-1}\right)} f\left(x^{(0:t-1)} \mid \mathcal{I}_{t-1}\right) \\ &\propto f\left(y^{(t)} \mid x^{(t)}\right) f\left(x^{(t)} \mid x^{(0:t-1)}\right) f\left(x^{(0:t-1)} \mid \mathcal{I}_{t-1}\right) \end{aligned}$$

Let q be the instrumental density such that it factorizes in the following way:

$$q\left(x_s^{(0:t)} \mid y^{(t)}, \mathcal{I}_{t-1}\right) = q\left(x_s^{(t)}, x_s^{(0:t-1)} \mid y^{(t)}, \mathcal{I}_{t-1}\right) q\left(x_s^{(0:t-1)}, \mathcal{I}_{t-1}\right)$$

We deduce that[56]:

$$\begin{aligned} w_s^{(t)} &\propto \frac{f\left(x_s^{(0:t)} \mid y^{(t)}, \mathcal{I}_{t-1}\right)}{q\left(x_s^{(0:t)} \mid y^{(t)}, \mathcal{I}_{t-1}\right)} \\ &\propto \frac{f\left(y^{(t)} \mid x_s^{(t)}\right) f\left(x_s^{(t)} \mid x_s^{(0:t-1)}\right) f\left(x_s^{(0:t-1)} \mid \mathcal{I}_{t-1}\right)}{q\left(x_s^{(t)}, x_s^{(0:t-1)} \mid y^{(t)}, \mathcal{I}_{t-1}\right)} \\ &= \frac{f\left(y^{(t)} \mid x_s^{(t)}\right) f\left(x_s^{(t)} \mid x_s^{(0:t-1)}\right)}{q\left(x_s^{(t)} \mid y^{(t)}, x_s^{(0:t-1)}, \mathcal{I}_{t-1}\right)} \cdot \frac{f\left(x_s^{(0:t-1)} \mid \mathcal{I}_{t-1}\right)}{q\left(x_s^{(0:t-1)} \mid y^{(t)}, \mathcal{I}_{t-1}\right)} \\ &= \frac{f\left(y^{(t)} \mid x_s^{(t)}\right) f\left(x_s^{(t)} \mid x_s^{(0:t-1)}\right)}{q\left(x_s^{(t)} \mid y^{(t)}, x_s^{(0:t-1)}, \mathcal{I}_{t-1}\right)} \cdot w_s^{(t-1)} \end{aligned} \tag{13.23}$$

The posterior density at time t can then be approximated as:

$$f\left(x^{(t)} \mid y^{(t)}\right) = \sum_{s=1}^{n_S} w_s^{(t)} \cdot \delta_x\left(x^{(t)} - x_s^{(t)}\right) \tag{13.24}$$

The previous computations lead to the sequential importance sampling (or SIS) algorithm:

1. at time t, we simulate $x_s^{(t)} \sim q\left(x_s^{(t)} \mid y^{(t)}, x_s^{(0:t-1)}, \mathcal{I}_{t-1}\right)$;
2. we update the weight $w_s^{(t)}$ using Equation (13.23);
3. we repeat steps 1 and 2 in order to obtain a sample of n_S particles;
4. we normalize the weights:

$$w_s^{(t)} \leftarrow \frac{w_s^{(t)}}{\sum_{s=1}^{n_S} w_s^{(t)}}$$

Remark 172 *With the SIS algorithm, the variance of the weights increases exponentially with the number of particles. This implies that some particles have negligible weights whereas others have large weights. This is why resampling techniques are generally added in order to reduce this phenomenon. Another method consists in simulating new (or auxiliary) particles at each time. Therefore, there exist several SMC algorithms (Arulampalam* et al.*, 2002; Doucet and Johansen, 2009): auxiliary particle filter (APF), generic particle filter (GPF), regularized particle filter (RPF), sampling importance resampling (SIR), etc.*

[56] Note that $y^{(t)} \in \mathcal{I}_t$.

An example We consider the following example:

$$\begin{cases} f\left(x^{(t)} \mid x^{(t-1)}\right) = \mathcal{N}\left(x^{(t)}; T\left(t, x^{(t-1)}\right), Q\right) \\ f\left(y^{(t)} \mid x^{(t)}\right) = \mathcal{N}\left(y^{(t)}; \kappa\left(x^{(t)}\right)^2, H\right) \end{cases}$$

The corresponding state space model is equal to:

$$\begin{cases} x^{(t)} = T\left(t, x^{(t-1)}\right) + \eta^{(t)} \\ y^{(t)} = \kappa\left(x^{(t)}\right)^2 + \varepsilon^{(t)} \end{cases} \tag{13.25}$$

where $\eta^{(t)} \sim \mathcal{N}(0, Q)$ and $\varepsilon^{(t)} \sim \mathcal{N}(0, H)$. We notice that the state space model is non-linear. The previous example has been extensively studied with the following specification (Carlin *et al.*, 1992; Kitagawa, 1996):

$$T(t, x) = \frac{x}{2} + \frac{25 \cdot x}{1 + x^2} + 8 \cdot \cos(1.2 \cdot t)$$

With the following values of parameters $\kappa = 1/20$, $Q = 1$ and $R = 10$, we simulate the model (13.25) and estimate $x^{(t)}$ by considering different particle filters. The likelihood and prior transition densities are given by $f\left(y^{(t)} \mid x^{(t)}\right)$ and $f\left(x^{(t)} \mid x^{(t-1)}\right)$. We assume that the instrumental density $q\left(x^{(t)} \mid y^{(t)}, x^{(t-1)}, \mathcal{I}_{t-1}\right)$ is equal to the transition density $f\left(x^{(t)} \mid x^{(t-1)}\right)$, meaning that the knowledge of $y^{(t)}$ does not improve the estimate of the state $x^{(t)}$. The particles are simulated according to the following scheme:

$$x_s^{(t)} = T\left(t, x_s^{(t-1)}\right) + \eta_s^{(t)}$$

where $\eta_s^{(t)} \sim \mathcal{N}(0, Q)$. In Figure 13.51, we report one simulation of the state space model. Recall that we observe $y^{(t)}$ and not $x^{(t)}$. The estimate $\hat{x}^{(t)}$ is equal to:

$$\hat{x}^{(t)} = \sum_{s=1}^{n_S} w_s^{(t)} \cdot x_s^{(t)}$$

where $x_s^{(t)}$ is the simulated value of $x^{(t)}$ for the s^{th} particle and $w_s^{(t)}$ is the importance weight given by Equation (13.23). For each simulation, we also calculate the root mean squared error:

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(x^{(t)} - \hat{x}^{(t)}\right)^2}$$

where $x^{(t)}$ is the true value of the state and $\hat{x}^{(t)}$ is the estimate. We report the probability density function of the RMSE statistic in Figure 13.52 when the number of particles is equal to 1 000. We notice that the SIR algorithm is better than the other SMC algorithms for this example. Figure 13.53 illustrates the convergence of the SIS algorithm with respect to the number of particles.

13.3.4 Quasi-Monte Carlo simulation methods

We consider the following Monte Carlo problem:

$$I = \int \cdots \int_{[0,1]^n} \varphi(x_1, \ldots, x_n)\, \mathrm{d}x_1 \cdots \mathrm{d}x_n$$

Let X be the random vector of independent uniform random variables. It follows that $I = \mathbb{E}[\varphi(X)]$. The Monte Carlo method consists in generating uniform coordinates in

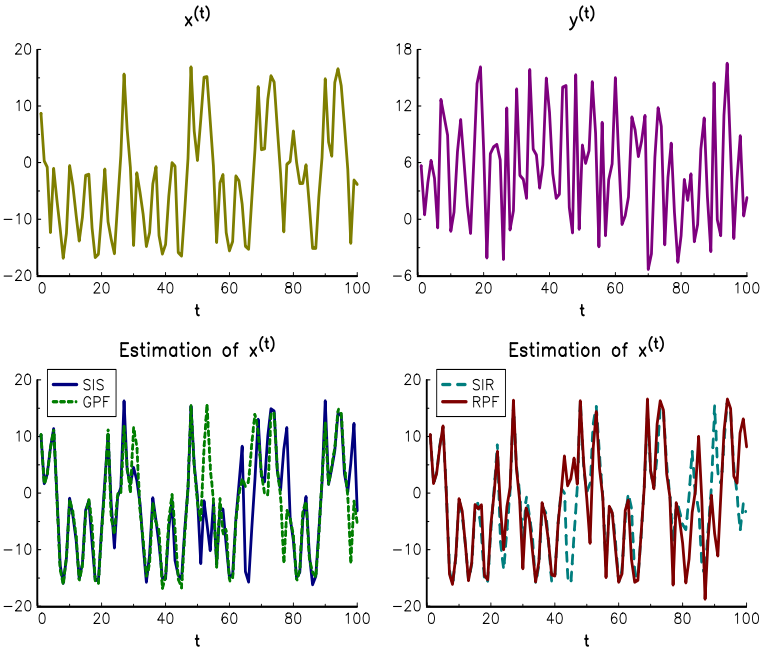

FIGURE 13.51: An example of a SMC run with 1 000 particles

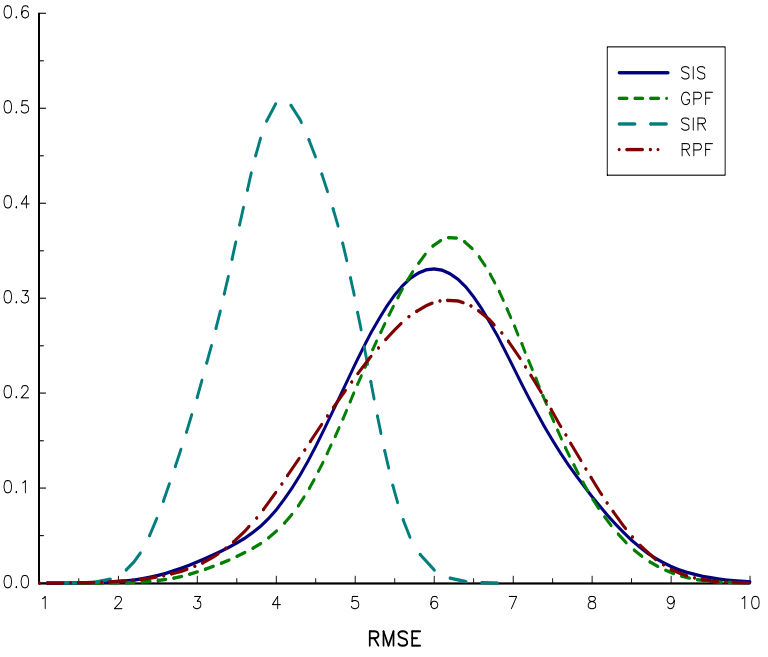

FIGURE 13.52: Density of the RMSE statistic for 1 000 particles

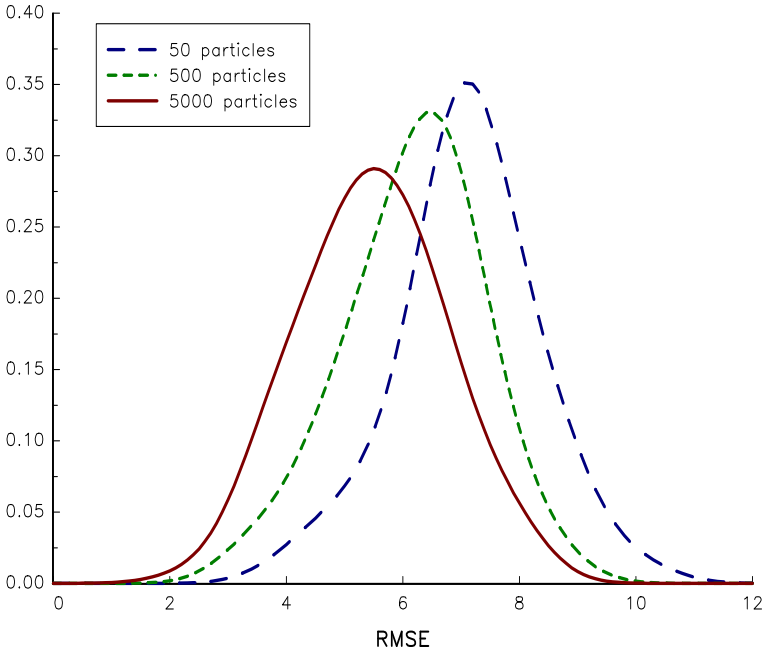

FIGURE 13.53: Density of the RMSE statistic for the SIS algorithm

the hypercube $[0,1]^n$. Quasi-Monte Carlo methods use non-random coordinates in order to obtain a more nicely uniform distribution. A low discrepancy sequence $\mathcal{U} = \{u_1, \ldots, u_{n_S}\}$ is then a set of deterministic points distributed in the hypercube $[0,1]^n$. Let us define the star discrepancy of $\mathcal{U}$ by $D^\star$:

$$D^\star_{n_s}(\mathcal{U}) = \sup_{x \in [0,1]^n} \left| \frac{1}{n_S} \sum_{s=1}^{n_S} \prod_{i=1}^{n} \mathbb{1}\{u_{i,s} \leq x_i\} - \prod_{i=1}^{n} x_i \right|$$

We could interpret $D^\star$ as the $\mathcal{L}_\infty$ norm between the theoretical continuous uniform distribution and the discrete uniform distribution generated by the low discrepancy sequence $\mathcal{U}$. We note that if $\mathcal{U}$ is really uniform, then $\lim_{n_S \to \infty} D^\star_{n_S}(\mathcal{U}) = 0$ for every dimension n. Moreover, Morokoff and Caflisch (1994) noticed that:

$$|I_{n_S} - I| \leq D^\star_{n_S}(\mathcal{U}) \cdot \mathcal{V}(f)$$

where $\mathcal{V}(f)$ is the Hardy-Krause variation of f. We could find low discrepancy sequences such that the error is of order $n_S^{-1} (\ln n_S)^n$ in probability (Morokoff and Caflisch, 1994). If we compare this bound with the order convergence $n_S^{-1/2}$ of MC, we notice that QMC is theoretically better than MC for small dimensions, but MC is better than QMC for high dimensions. However, in practice, it appears that QMC could be more accurate than MC even for large dimension n.

Glasserman (2003) reviewed different quasi-random sequences. The most known are the Halton, Sobol and Faure sequences and corresponding numerical codes are available in different programming languages (Press *et al.*, 2007). The techniques to generate these sequences are based on number theory. For example, the Halton sequence is based on the p-adic expansion of integers $n = d_k p^k + \cdots + d_1 p + d_0$ and the radical-inverse function $\varrho_b(n) = \sum_{i=0}^{k} d_k p^{-(i+1)}$ where $d_i \in \{0, \ldots, p-1\}$ for $i = 0, \ldots, k$. The d-dimensional

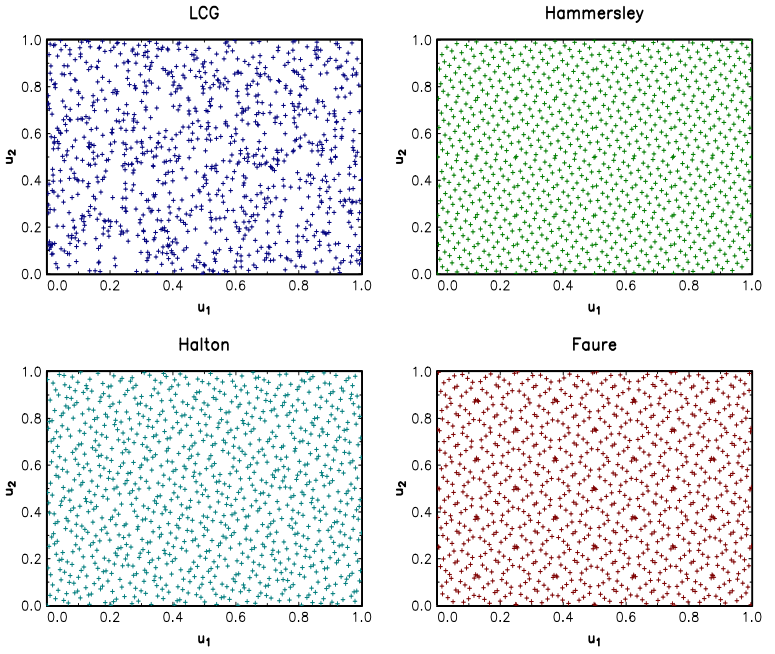

FIGURE 13.54: Comparison of different low discrepancy sequences

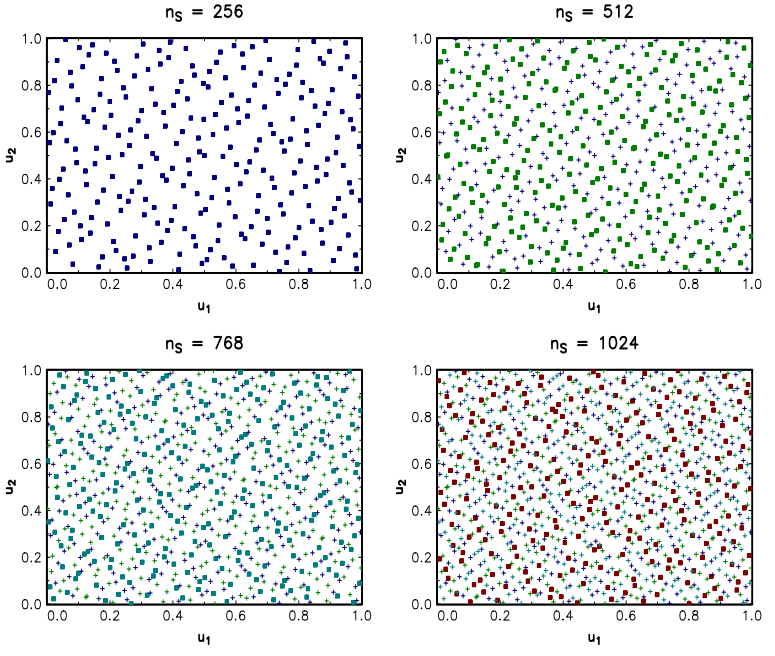

FIGURE 13.55: The Sobol generator

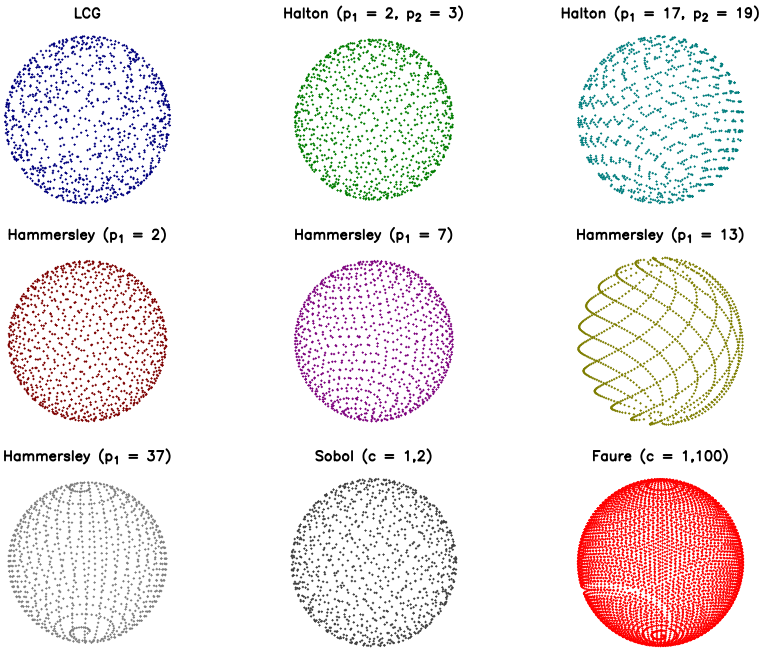

FIGURE 13.56: Quasi-random points on the unit sphere

TABLE 13.8: Pricing of the spread option using quasi-Monte Carlo methods

n_S	10^2	10^3	10^4	10^5	10^6	5×10^6
LCG (1)	4.3988	5.9173	5.8050	5.8326	5.8215	5.8139
LCG (2)	6.1504	6.1640	5.8370	5.8219	5.8265	5.8198
LCG (3)	6.1469	5.7811	5.8125	5.8015	5.8142	5.8197
Hammersley (1)	32.7510	26.5326	21.5500	16.1155	9.0914	5.8199
Hammersley (2)	32.9082	26.4629	21.5465	16.1149	9.0914	5.8199
Halton (1)	8.6256	6.1205	5.8493	5.8228	5.8209	5.8208
Halton (2)	10.6415	6.0526	5.8544	5.8246	5.8208	5.8207
Halton (3)	8.5292	6.0575	5.8474	5.8235	5.8212	5.8208
Sobol	5.7181	5.7598	5.8163	5.8190	5.8198	5.8198
Faure	5.7256	5.7718	5.8157	5.8192	5.8197	5.8198

Halton sequence $\varrho = \{\varrho_n\}$ is then defined by $\varrho_n = (\varrho_{p_1}(n), \ldots, \varrho_{p_d}(n))$ where $p_1, \ldots, p_d$ are integers that are greater than one and pairwise relatively prime. We represent this sequence when $d = 2$ and $n_S = n = 1024$, and compare it to LCG random variates and Hammersley and Faure sequences in Figure 13.54. The underlying idea of QMC is to add the new points not randomly, but between the existing points. For example, we have added 256 points in the Sobol sequence[57] in each panel of Figure 13.55. Finally, we report the projection of different low discrepancy sequences on the unit sphere in Figure 13.56. We notice that we can generate some '*hole area*'.

Example 160 *We consider a spread option whose payoff is equal to* $(S_1(T) - S_2(T) - K)^+$. *The price is calculated using the Black-Scholes model, and the following parameters:*

[57] The new points correspond to a square symbol.

$S_1(0) = S_2(0) = 100$, $\sigma_1 = \sigma_2 = 20\%$, $\rho = 50\%$ and $r = 5\%$. The maturity T of the option is set to one year, whereas the strike K is equal to 5. We estimate the option price with several QMC methods and different number of simulations n_S. Results are given in Table 13.8. We consider three seed values for the LCG pseudorandom sequences. In the case of Hammersley sequences, we use $p_1 = 2$ (sequence 1) and $p_1 = 7$ (sequence 2). We also use three Halton sequences based on the following values of (p_1, p_2): $(2,3)$, $(17,19)$ and $(2,19)$. Finally, we consider Sobol and Faure sequences when the dimension is equal to 2. For simulating Gaussian random variates, we use the Box-Muller method. The true price of the spread option is equal to 5.8198. We notice that only the Sobol and Faure generators have converged to this price when the number of simulations is equal to one million.

13.4 Exercises

13.4.1 Simulating random numbers using the inversion method

1. Propose an algorithm to simulate random variates for the following distribution functions:
   (a) the generalized extreme value distribution $\mathcal{GEV}(\mu, \sigma, \xi)$;
   (b) the log-normal distribution $\mathcal{LN}(\mu, \sigma^2)$;
   (c) the log-logistic distribution $\mathcal{LL}(\alpha, \beta)$.

2. When we model operational risk losses, we are interested in the conditional random variable $L = X \mid X \geq H$ where H is a given threshold.
   (a) How can we simulate L if we consider a random number generator of X?
   (b) Let $\mathbf{F}_X$ be the distribution function of X. Give the conditional distribution $\mathbf{F}_L$.
   (c) Find the inverse function $\mathbf{F}_L^{-1}$ and propose an algorithm to simulate L.
   (d) Compare the two algorithms in terms of efficiency.
   (e) Apply algorithms (a) and (c) to the log-normal distribution $\mathcal{LN}(\mu, \sigma^2)$. We assume that $\mu = 7$, $\sigma = 2.3$ and $H = \$50\,000$. Simulate 100 random numbers[58] and draw the scatterplot between the uniform random numbers u_i and the random variates L_i.

3. We consider the extreme order statistics $X_{1:n} = \min(X_1, \ldots, X_n)$ and $X_{n:n} = \max(X_1, \ldots, X_n)$.
   (a) How can we simulate $X_{1:n}$ and $X_{n:n}$ if we have a random generator for X_i?
   (b) Calculate the distribution functions $\mathbf{F}_{1:n}$ and $\mathbf{F}_{n:n}$. Deduce an efficient algorithm to simulate $X_{1:n}$ and $X_{n:n}$.
   (c) Using the previous algorithm, simulate 1000 random numbers of $X_{1:50}$ and $X_{50:50}$ when $X_i \sim \mathcal{N}(0,1)$.

[58] We can use the Lewis-Goodman-Miller generator with a seed equal to 123 456.

13.4.2 Simulating random numbers using the transformation method

1. We consider the random variable X, whose probability density function is given by:

$$f(x) = \frac{\beta^\alpha x^{-\alpha-1} e^{-\beta/x}}{\Gamma(\alpha)}$$

Calculate the density function of $Y = 1/X$. Deduce that X follows the inverse-gamma distribution $\mathcal{IG}(\alpha, \beta)$. Find an algorithm to simulate X.

2. Let $X \sim \mathcal{G}(\alpha, \beta)$ be a gamma distributed random variable.

   (a) Show that the case $\alpha = 1$ corresponds to the exponential distribution with parameter $\lambda = \beta$:

$$\mathcal{G}(1, \beta) = \mathcal{E}(\beta)$$

   Deduce an algorithm to simulate $\mathcal{G}(1, \beta)$.

   (b) When α is an integer and is equal to n, show that $X \sim \mathcal{G}(n, \beta)$ is the sum of n independent exponential random variables with parameter β. Deduce an algorithm to simulate $\mathcal{G}(n, \beta)$.

3. Let $X \sim \mathcal{B}(\alpha, \beta)$ be a beta distributed random variable.

   (a) We note $Y \sim \mathcal{G}(\alpha, \delta)$ and $Z \sim \mathcal{G}(\beta, \delta)$ two independent gamma distributed random variable. Show that[59]:

$$X = \frac{Y}{Y + Z}$$

   (b) Deduce an algorithm to simulate X.

4. The polar method considers the random vector (X, Y) defined by:

$$\begin{cases} X = R \cdot \cos\Theta \\ Y = R \cdot \cos\Theta \end{cases}$$

where R and Θ are two independent random variables. We assume that $R \sim \mathbf{F}_R$ and $\Theta \sim \mathcal{U}_{[0,2\pi]}$.

   (a) Show that the joint density of (X, Y) is equal to:

$$f_{X,Y}(x, y) = \frac{f_R\left(\sqrt{x^2 + y^2}\right)}{2\pi\sqrt{x^2 + y^2}}$$

   Deduce the expression of the density function $f_X(x)$ of X.

   (b) We assume that $R = \sqrt{2 \cdot E}$ where $E \sim \mathcal{E}(1)$.

      i. Show that $f_R(r) = re^{-r^2/2}$.

      ii. Calculate $f_X(x)$.

      iii. Deduce the Box-Muller algorithm to simulate normal distributed random variables.

[59] Hint: consider the change of variables $X = Y/(Y + Z)$ and $S = Y + Z$, and calculate the joint density function $f_{X,S}$.

(c) Bailey (1994) proposed to simulate the Student's t_ν distribution with the polar method. For that, he considered the distribution:

$$\mathbf{F}_R(r) = 1 - \left(1 + \frac{r^2}{\nu}\right)^{-\nu/2}$$

where $r \geq 0$ and $\nu > 0$.

i. Calculate $f_R(r)$.

ii. Show that:

$$f_{X,Y}(x,y) = \frac{1}{2\pi}\left(1 + \frac{x^2+y^2}{\nu}\right)^{-\nu/2-1}$$

iii. Find the expression[60] of $f_X(x)$. Deduce that X is a Student's t random variable with ν degrees of freedom.

iv. Find an algorithm to simulate R.

v. Deduce an algorithm to simulate Student's t_ν random variables.

vi. What is the main difference with the Box-Muller algorithm?

13.4.3 Simulating random numbers using rejection sampling

1. We consider the beta distributed random variable $X \sim \mathcal{B}(\alpha, \beta)$. We assume that $\alpha \geq 1$ and $\beta \geq 1$.

   (a) We use the proposal density function $g(x) = 1$. Calculate the function $h(x)$ defined as follows:

   $$h(x) = \frac{f(x)}{g(x)}$$

   Show that $h(x)$ achieves its maximum at the point:

   $$x^\star = \frac{\alpha - 1}{\alpha + \beta - 2}$$

   Deduce the value of c that maximizes the acceptance ratio.

   (b) Plot the functions $f(x)$ and $cg(x)$ for the following parameters (α, β): $(1.5, 1.5)$, $(3, 2)$, $(1, 8)$ and $(5, 7)$. Comment on these results.

   (c) Propose an algorithm to simulate $\mathcal{B}(\alpha, \beta)$ when $\alpha \geq 1$ and $\beta \geq 1$.

2. We consider the beta distributed random variable $X \sim \mathcal{B}(\alpha, \beta)$. We assume that $\alpha < 1$ and $\beta \geq 1$.

   (a) We use the proposal density function $g(x) = \alpha x^{\alpha-1}$. Find the value of c that maximizes the acceptance ratio.

   (b) Give an algorithm to simulate the random variable $X \sim \mathbf{G}$.

   (c) Give an algorithm to simulate $\mathcal{B}(\alpha, \beta)$ when $\alpha < 1$ and $\beta \geq 1$.

3. We consider the standard Gaussian random variable $X \sim \mathcal{N}(0, 1)$.

[60] Hint: consider the change of variable $u = \left(1 + \frac{y^2}{\nu + x^2}\right)^{-1}$.

(a) We use the Laplace distribution as the proposal distribution:

$$g(x) = \frac{1}{2}e^{-|x|}$$

Calculate $\mathbf{G}(x)$ and $\mathbf{G}^{-1}(x)$. Give an algorithm to simulate the Laplace distribution.

(b) Find the value of c that maximizes the acceptance ratio. Draw the functions $f(x)$ and $cg(x)$.

(c) Deduce the acceptance-rejection algorithm.

4. We consider the standard gamma random variable $X \sim \mathcal{G}(\alpha)$ where $\alpha \geq 1$.

(a) We use the Cauchy distribution as the proposal distribution:

$$g(x) = \frac{1}{\pi(1+x^2)}$$

Show that:

$$\frac{f(x)}{g(x)} \leq \frac{2\pi}{\Gamma(\alpha)}x^{\alpha+1}e^{-x}$$

Find the value of c that maximizes the acceptance ratio.

(b) We use the Student's t distribution with 2 degrees of freedom as the proposal distribution. Calculate the analytical expression of $\mathbf{G}(x)$. Deduce an algorithm to simulate X.

(c) In the case (b), Devroye (1986) showed that:

$$f(x) \leq cg(x)$$

where:

$$c = \frac{\sqrt{3(4\alpha-1)}}{\Gamma(\alpha)}(\alpha-1)^{\alpha-1}e^{-(\alpha-1)}$$

What is the most efficient method between algorithms (a) and (b)?

5. We consider a discrete random variable X with a finite number of states x_k where $k = 1, \ldots, K$. We note $p(k) = \Pr\{X = x_k\}$ its probability mass function.

(a) We consider the following proposal distribution:

$$q(k) = \frac{1}{K}$$

Find the value of c that maximizes the acceptance ratio.

(b) We consider the following distribution:

x_k	0	1.5	3.3	5.6	8.9
$p(k)$	10%	20%	40%	20%	10%

Simulate 1 000 random numbers using the acceptance-rejection algorithm and draw the histogram of accepted and rejected values. Comment on these results.

13.4.4 Simulation of Archimedean copulas

We recall that an Archimedean copula has the following expression:

$$\mathbf{C}(u_1, u_2) = \varphi^{-1}(\varphi(u_1) + \varphi(u_2))$$

where φ is the generator function.

1. Retrieve the Genest-MacKay algorithm to simulate Archimedean copulas.
2. We assume that $\varphi(u) = (-\ln u)^{\theta}$ with $\theta \geq 1$. Find the corresponding copula.
3. Calculate the conditional distribution $\mathbf{C}_{2|1}$ associated to the previous Archimedean copula. Deduce an algorithm to simulate it.
4. We consider the Frank copula defined as follows:

$$\mathbf{C}(u_1, u_2) = -\frac{1}{\theta} \ln \left(1 + \frac{\left(e^{-\theta u_1} - 1\right)\left(e^{-\theta u_2} - 1\right)}{e^{-\theta} - 1} \right)$$

   where $\theta \in \mathbb{R}$. Calculate the conditional distribution $\mathbf{C}_{2|1}$ and deduce an algorithm to simulate this copula.
5. The Ali-Mikhail-Haq family of copulas is given by:

$$\mathbf{C}(u_1, u_2) = \frac{u_1 u_2}{1 - \theta(1 - u_1)(1 - u_2)}$$

   where θ lies in $[-1, 1]$. Verify that the generator of this family is:

$$\varphi(u) = \ln \left(\frac{1 - \theta(1 - u)}{u} \right)$$

6. Simulate 5 random vectors of the Gumbel-Hougaard ($\theta = 1.8$), Frank ($\theta = 2.1$) and Ali-Mikhail-Haq copulas ($\theta = 0.6$) by using the following uniform random variates:

v_1	0.117	0.607	0.168	0.986	0.765
v_2	0.498	0.400	0.269	0.892	0.109

13.4.5 Simulation of conditional random variables

Let $Z \sim \mathcal{N}(\mu_z, \Sigma_{z,z})$ be a Gaussian random vector of dimension n_z. We consider the partition $Z = (X, Y)$ where $n_x + n_y = n_z$, $\mu_z = (\mu_x, \mu_y)$ and:

$$\Sigma_{zz} = \begin{pmatrix} \Sigma_{x,x} & \Sigma_{x,y} \\ \Sigma_{y,x} & \Sigma_{y,y} \end{pmatrix}$$

1. Let T be the random vector Y given that $X = x^\star$. Give the distribution of T. Deduce an algorithm to simulate T.
2. We consider the random vector $\tilde{T}$ defined by:

$$\tilde{T} = Y - \Sigma_{y,x} \Sigma_{x,x}^{-1} (X - x^\star)$$

   Show that $\tilde{T} = T$. Deduce an algorithm to simulate T.

3. How can we simulate the Gaussian random vector Z without using the Cholesky decomposition?

4. We assume that the vector of means is $(1, 2, 3)$, the vector of standard deviations is $(1, 0.5, 5)$ and the correlation matrix is:

$$\mathbb{C} = \begin{pmatrix} 1.00 & & \\ 0.50 & 1.00 & \\ 0.20 & 0.30 & 1.00 \end{pmatrix}$$

Apply the algorithm described in Question 3 by using the following independent Gaussian random variates $\mathcal{N}(0, 1)$:

u_1	-1.562	-0.563	-0.573	-0.596	0.984
u_2	0.817	0.845	0.872	-1.303	-0.433
u_3	-0.670	0.126	0.884	-0.918	-0.052

13.4.6 Simulation of the bivariate Normal copula

Let $X = (X_1, X_2)$ be a standard Gaussian vector with correlation ρ. We note $U_1 = \Phi(X_1)$ and $U_2 = \Phi(X_2)$.

1. We note Σ the matrix defined as follows:

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

Calculate the Cholesky decomposition of Σ. Deduce an algorithm to simulate X.

2. Show that the copula of (X_1, X_2) is the same that the copula of the random vector (U_1, U_2).

3. Deduce an algorithm to simulate the Normal copula with parameter ρ.

4. Calculate the conditional distribution of X_2 knowing that $X_1 = x$. Then show that:

$$\Phi_2(x_1, x_2; \rho) = \int_{-\infty}^{x_1} \Phi\left(\frac{x_2 - \rho x}{\sqrt{1-\rho^2}}\right) \phi(x)\, \mathrm{d}x$$

5. Deduce an expression of the Normal copula.

6. Calculate the conditional copula function $\mathbf{C}_{2|1}$. Deduce an algorithm to simulate the Normal copula with parameter ρ.

7. Show that this algorithm is equivalent to the Cholesky algorithm found in Question 3.

13.4.7 Computing the capital charge for operational risk

We assume that the mapping matrix contains two cells. For each cell, the aggregate loss S_k is defined as:

$$S_k = \sum_{i=1}^{N_k} X_{k,i}$$

where $N_k \sim \mathcal{P}(\lambda_k)$ is the number of losses and $X_{k,i} \sim \mathcal{LN}(\mu_k, \sigma_k^2)$ are the individual losses. The total loss for the bank is then equal:

$$L = S_1 + S_2$$

We calculate the capital-at-risk $\mathrm{CaR}(\alpha)$ for different confidence levels: 90%, 95%, 99% and 99.9%. For that, we use one million simulations.

1. We consider the first cell $k = 1$ and we assume that $\lambda_1 = 50$, $\mu_1 = 7$ and $\sigma_1 = 1.5$. Using 100 replications, calculate the mean and standard deviation of the estimator $\widehat{\mathrm{CaR}}_1(\alpha)$. Do you think that one million simulations is sufficient?
2. Same question for the second cell $k = 2$ if we assume that $\lambda_2 = 100$, $\mu_2 = 5.5$ and $\sigma_2 = 1.8$.
3. Represent the probability density function of $\ln L$ when the aggregate losses S_1 and S_2 are independent and perfectly dependent. Calculate the diversification ratio when we assume that S_1 and S_2 are independent.
4. We assume that the dependence function $\mathbf{C}\langle S_1, S_2\rangle$ is a Normal copula with parameter ρ. Calculate the capital-at-risk of the bank for the following values of ρ: 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% and 100%. Compare these estimates with those obtained with a Gaussian approximation.
5. Same question if the dependence function between S_1 and S_2 is a t_4 copula.
6. Same question if the dependence function between S_1 and S_2 is a t_1 copula.
7. Comment on these results.

13.4.8 Simulating a Brownian bridge

We consider a Brownian bridge $B(t)$ such that $s \leq t \leq u$, $W(s) = w_s$ and $W(u) = w_u$.

1. Find the distribution of the random vector $(W(s), W(t), W(u))$.
2. Calculate the conditional distribution of $W(t)$ given that $W(s) = w_s$ and $W(u) = w_u$.
3. Deduce an algorithm to simulate $B(t)$.

13.4.9 Optimal importance sampling

We consider the estimation of the probability $p = \Pr\{X \geq c\}$ when $X \sim \mathcal{N}(0,1)$.

1. We note $\hat{p}_{\mathrm{MC}}$ the MC estimator of p for one simulation. Calculate $\mathbb{E}[\hat{p}_{\mathrm{MC}}]$ and $\mathrm{var}(\hat{p}_{\mathrm{MC}})$. What is the probability distribution of $\hat{p}_{\mathrm{MC}}$?
2. Let $\mathcal{N}(\mu, \sigma^2)$ be the importance sampling distribution. Give the expression of the IS estimator $\hat{p}_{\mathrm{IS}}$ for one simulation. Calculate $\mathbb{E}[\hat{p}_{\mathrm{IS}}]$ and $\mathrm{var}(\hat{p}_{\mathrm{IS}})$. What do you notice about the probability distribution of $\hat{p}_{\mathrm{IS}}$?
3. We assume that $c = 3$. Calculate $\mathrm{var}(\hat{p}_{\mathrm{MC}})$. Draw the relationship between μ and $\mathrm{var}(\hat{p}_{\mathrm{IS}})$ when σ is respectively equal to 0.8, 1, 2 and 3. Find the optimal value of μ. What hypothesis can we make?
4. We assume that σ is equal to 1. Find the first-order condition if we would like to select the optimal important sampling scheme. Draw the relationship between c and the optimal value of μ. Deduce an heuristic approach for defining a good IS scheme.