11 Monte Carlo Simulations

Having recognized the fact that prices of financial instruments can be calculated as discounted future expectations (with respect to a risk-neutral probability measure), the idea of calculating such expectations by simulating the (stochastic) evolution of the underlyings several times and subsequently averaging the results somehow is not far removed. In fact, this relatively simple idea is widely used and is successful even in the valuation of very exotic options for which other methods are either too complicated or completely unsuitable, the only requirement being the availability of sufficient computation time. Before proceeding with financial *applications* of Monte Carlo techniques, we begin with a presentation of the technique itself.

If random events occur often enough, they can be used to answer diverse questions statistically. This has long been common knowledge in science and we have seen a vast increase in applications with the advance of modern computers, since computers suddenly made it possible to generate "random" events cheaply and in large numbers, or in the language of the specialist, to *simulate* them. Ever since, *computer simulations* have been indispensable in science, and since lately also in the modern financial world. Since generating "random" events lies at the core of such simulations, the name *Monte Carlo simulation* has become accepted despite the fact that the method's namesake city in Monaco could never generate as many random events as are sometimes necessary in practice, even if all the casinos in Monte Carlo were open non-stop for business every day for a million years. From this point of view, the computer can far outperform the roulette table.

H.-P. Deutsch, M. W. Beinker, *Derivatives and Internal Models*, Finance and Capital Markets Series, https://doi.org/10.1007/978-3-030-22899-6_11

With simulations, it has become possible to solve problems for which classical solutions fail, e.g. if the dimensionality of the problem is to high (regarding for example most equity basket products with three or more different stocks in the basket), or if the problem at hand could not be formulated in terms of a partial differential equation (e.g. the multi-factor LIBOR market model). Use of Monte-Carlo simulations, on the other hand, allow for the direct simulation of stochastic differentials. Even complex path dependencies with the final pay off depending highly on the evolution path of the underlying over time can be modeled by means of Monte Carlo methods without difficulty, since the various paths are simulated directly anyway. Lattice methods (trees, finite differences) can be used to solve path-dependent problems only in some special, simple cases or with great computational effort.

In addition, computer simulations allow to perform a what-if-analysis, with which in short time many different scenarios could be simulated and analyzed. The scenarios to be simulated could be given as fixed parameter sets (*static simulation*, without any stochastic parameters, and are therefore no Monte Carlo simulations) or randomly synthesized by application of a set of rules (*dynamic Monte Carlo simulation*). The recently becoming increasingly popular stress tests are nothing else then computer simulations, too. Finally, computer simulations are fairly easy to implement and understood.

This advantages do have a price: computer simulations require lots of computation time, and especially for low-dimensional problems, this is a significant disadvantage with respect to lattice methods. Also, it is very difficult and time intensive to calculate sensitivities with required accuracy. Therefore, MC simulations are often the method of last resort, i.e. MC simulations should in general only be used if other alternative methods fail.

Assumptions

A Monte Carlo simulation in its most general form consistent with arbitrage-free pricing requires only the assumptions eliminating arbitrage opportunities, i.e., Assumptions 1, 2, 3, 4 and 5 from Chap. 4. In addition, at first, we constrain our considerations to European options only. As will be shown later, the valuation of American Options with Monte Carlo methods requires significant extra effort.

The Monte Carlo method presented in the following is based on the random walk equation 2.17 with constant drift and volatility. Hence, because of Eq. 9.25 for pricing in a risk-neutral world, constant yields must also be assumed. This means that for the method presented, the additional

Assumptions 7, 9, 11 and 12 from Chap. 4 are also made. The assumptions of constant interest rates and volatilities naturally imply that interest rates and volatilities are non-stochastic, i.e., Assumptions 8 and 10 hold as well. Generalization to time-dependent parameters is possible without problems, though.

These extensive assumptions are made to allow a clear presentation of the material but are, in principle, not necessary for performing Monte Carlo simulations. For example, not only the underlying price, but also the volatility can be simultaneously simulated as stochastic processes (for instance also as a random walk). Since then two of the parameters involved are stochastic processes, the random walk occurs now in two dimensions. Of course, the volatility of the volatility and also the drift of the volatility would be required as parameters in such a situation. The simulation then proceeds as follows: first, a value for the volatility is simulated. Using this simulated volatility as a parameter, the next step for the underlying price is simulated. The two random walks are thus not independent of one another since the random walk describing the volatility affects that of the price. In reality, it can often be observed that the converse holds as well, i.e., the price of the underlying has an influence on the volatility (low volatility for rising prices and high volatility for falling prices). This effect can be incorporated into the simulation using a special form for the drift of the volatility random walk, which depends on the underlying price, for example as "volatility-drift = $rdt/S(t)$".

When simulating interest rates for pricing caps or floors, for example, Eq. 2.17 can naturally be replaced by a more complex process corresponding to a term structure model such as Heath-Jarrow-Morton, Ho-Lee, Hull-White, etc. Mean reversion, for example, intuitively corresponds to a random walk under the influence of an external force which has the effect of "driving" the random walk back to an asymptotic mean value.

11.1 A Simple Example: The Area of a Disk

First, we will make some effort to understand clearly the essence of the Monte Carlo simulation, before the method will be applied to an actual problem. Therefore, at this point we present a very simple example of the idea behind the Monte Carlo method before entering into a discussion of applications in the financial world. As known from elementary mathematics, the area of a disk of radius R is πR^2, the product of the square of the radius with the constant π, already known to the ancient Egyptians: $\pi = 3.14159...$ A disk with a radius of one meter, $R = 1m$, thus has an area of $\pi \times (1m)^2 = 3.14159m^2$.

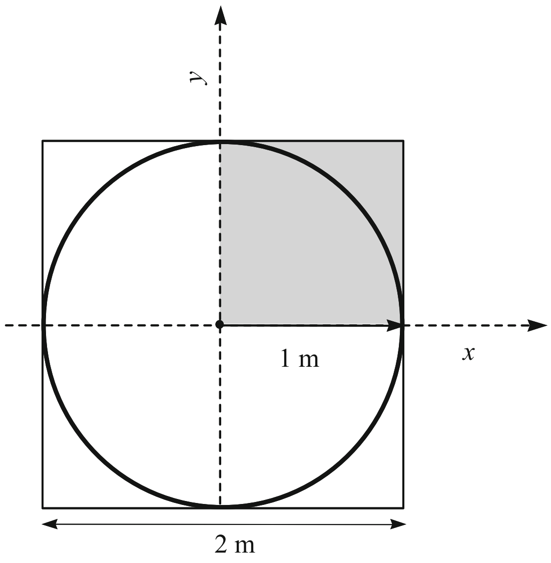

Fig. 11.1 A disk with diameter $2m$ in a square with sides of length $2m$

Supposing we had never heard of the number π, this fact can be ascertained with the help of random events. To do so, we simply place square box with sides of length $2m$ containing a "pie dish" as shown in Fig. 11.1 out in the rain. The pie dish represents the disk of radius $1m$ whose area is to be determined. We know that the square has an area of $2m \times 2m = 4m^2$. The random events are the falling raindrops. Assuming that the raindrops fall evenly on the square, then the area of the disk can be given by

$$\text{disk area} = \frac{\text{number of raindrops falling on the disk}}{\text{number of raindrops falling on the entire square}} \times 4m^2 \,.$$

In this way, the area of a disk can be determined with the help of random events.

The same "Experiment" admits another interpretation. If the equation "disk area = πR^2" is known, then the value of π can be determined:

$$\begin{aligned}\pi &= \frac{\text{disk area}}{R^2} = \frac{\text{disk area}}{1m^2} \\ &= 4 \times \frac{\text{number of raindrops falling on the disk}}{\text{number of raindrops falling on the entire square}} \,.\end{aligned}$$

Thus, with the help of randomly falling raindrops, the value of the natural constant π has been determined.

The primary application of Monte Carlo methods—in particular in the financial world—is the calculation of integrals. For example, the price of an option can be expressed as the integral of its payoff profile with respect to an appropriate probability measure associated to the price of its underlying at maturity. Making use of our disc-example, we give a third interpretation of the experiment described above to illustrate how random events can be used to calculate an integral: as is known from elementary mathematics, a semi-circle can be represented as the graph of a function. Choosing the x- and y-axis as shown in Fig. 11.1, the ancient *Pythagorean Theorem* says that the points (x, y) on the circle all satisfy $x^2 + y^2 = R^2 = 1$, thus the upper semi-circle is the graph of the function $y = \sqrt{1 - x^2}$. The lower semi-circle is the graph of the function $y = -\sqrt{1 - x^2}$. As is well know from any introduction to mathematical analysis, integrating a function gives the area under the curve given by the graph of that function. Since the upper semi-circle is represented by the graph of the function $y = \sqrt{1 - x^2}$, the area of the half-disk can be determined by integrating:

$$\begin{aligned}\int_{-1}^{1} \sqrt{1 - x^2}dx &= \frac{1}{2}\text{Area of the disk} \\ &= 2\frac{\text{Number of raindrops falling on the disc}}{\text{Number of raindrops falling on the entire square}} .\end{aligned}$$

Thus, we have calculated this integral with the help of randomly falling raindrops.

Note that the quantity we want to calculated using random events is by no means random itself: the area of a disk of radius $1m^2$ never changes nor does the value of π or the value of the above integral. We could ask the question: does rain know anything about geometry and circles? Or about the natural constant π? Or about integration? Probably not. However, this information can be obtained if we are clever enough to ask the right questions! The only condition, the randomly falling raindrops had to meet, was that they fall evenly, or more precisely, that the probability for them to fall on any particular point in the square is exactly the same as to fall on any other point in the square. For example, on a square field with an area of 1000 square miles, it might be raining in some places and sunny in others. Such a field would be completely unsuitable for our experiment.

The "machine" used to generate the random events, whether the rain or a computer, is completely irrelevant. The events merely have to be "sufficiently random" and their probability distribution must be known. If this is the case, calculations can be performed with an accuracy which is limited solely by the machine's capacity to produce the random events: according to the laws of statistics, the accuracy of results improves as the number of random events involved in the simulation increases (see Sect. 31.2).

How would the above experiment have been conducted with a computer? Nowadays, most programming languages such as Pascal, Fortran, C, C++, etc. and the common spread sheet programs, such as Microsoft Excel (or Visual Basic) or Lotus 1-2-3, are equipped with *random number generators*. These are (small) programs or functions which usually generate uniformly distributed *random numbers* between 0 and 1. Simulating a random event such as a falling raindrop with a computer is accomplished by generating *two* such random numbers Z_1 and Z_2 to simulate the coordinates (one x- and one y-coordinate) of the point on which the raindrop falls. Since the simulated random numbers are all between 0 and 1 the generator producing such coordinates simulates raindrops falling only in the shaded area in Fig. 11.1. To simulate raindrops falling on the entire square, the following transformation must be made:

$$x = 2Z_1 - 1 \, , \quad y = 2Z_2 - 1 \, .$$

Because this transformation is linear, x and y are uniformly distributed random variables, as are Z_1 and Z_2. Thus, the coordinates of a raindrop in the square have been determined. The simulation of the random event is complete and we can continue with the evaluation. As in the case of raindrops, two events must be counted:

- The total number of "raindrops" falling in the square. This, however, is exactly the number of simulated random events, since no coordinates were generated which lie outside of the square (we make no unnecessary simulations). This means that this counter will be increased by one after each random event has been simulated.
- The number of "raindrops" falling within the circle. According to *Pythagoras*, these are the random events whose coordinates x and y satisfy the inequality $x^2 + y^2 \leq 1$. This means that after each random event, the counter is only increased by one if the generated event satisfies this inequality.

If, for example, 10,000 events were simulated (this requires the generation of 20,000 random numbers, one for each coordinate), of which, for example, 7851 satisfy the above inequality, we obtain

$$\text{Area of the disk} \approx 0,7851 * 4m^2 = 3,1404m^2$$

$$\pi \approx 0,7851 * 4 = 3,1404$$

$$\int_{-1}^{1} \sqrt{1-x^2}\, dx \approx 0,7851 * 2 = 1,5702\ .$$

Of course, this result is not exact, but the statistical error involved can be determined as described in Sect. 31.2. In principle, this error can be reduced almost arbitrarily by increasing the number of simulated random events, as long as computer capacity allows it.

11.2 The General Approach to Monte Carlo Simulations

We can extract the general procedure for conducting a Monte Carlo simulation from the above example which will be summarized here to provide the reader with a "recipe" for performing such simulations. Each step will be explained by reference to its corresponding step in the above disk experiment as well as to the simulation of a random walk.

1. **Generate the random numbers required for a Monte Carlo step, usually from a uniform distribution between 0 and 1.**
   In the disk experiment, a Monte Carlo step corresponds to the falling of a simulated "raindrop". Two random numbers were required. To simulate a random walk, a Monte Carlo step represents a step in the random walk. One random number is required for each dimension of the space in which the random walk occurs.
2. **Transform the random numbers to generate numbers according to a desired distribution.**
   In the disk experiment, the uniformly distributed random numbers generated between 0 and 1 were transformed to yield uniformly distributed random numbers between -1 and 1. For a random walk, they are trans-

formed to normally distributed random numbers. Section A.5 presents methods for doing this.

3. **Perform a Monte Carlo step using the random numbers generated.**
   In the case of the disk, this corresponds to the falling of a "raindrop". For the random walk, it consists of adding one time step to the random walk.
4. **Repeat steps 1, 2 and 3 until the system reaches the state required for the proposed investigation.**
   For a random walk consisting of n steps, the required state is attained when n Monte Carlo simulations have been carried out. For the disk, the evaluation can take place after each single Monte Carlo step.
5. **Measurements: measure the system variables of interest.**
   In the case of the disk, we measure the number of "raindrops" falling within the disk. For a random walk, for example, the "end-to-end" distance could be measured.
6. **Repeat steps 1, 2, 3, 4 and 5 until enough systems for a statistical analysis have been generated**.
   For the disk, many "raindrops" must be simulated; for the price evolution of an underlying, many random walks must be generated.
7. **Final Analysis: compute the means of the measured variables and determine the statistical error.**
   For the disk, this was the ratio of the number of raindrops falling within the disk to the total number simulated. For a random walk it could be, for example, the mean length of all measured end-to-end vectors (or the square of their Euclidean norm).

11.3 Monte Carlo Simulation of Risk Factors

11.3.1 Simulation of the Evolution of a *Single* Risk Factor

On the basis of Eq. 2.17, the time interval from t to T is divided into n intervals of length δt where n equals the number of steps. The price of a risk factor is to be simulated as a random walk over this time interval. The random walk equation 2.17 (or its equivalent form 2.23) holds for infinitesimal changes in $\ln(S)$ occurring in an infinitesimally small time span dt. The time step δt used for the simulation is not infinitesimally small, however. Therefore we do not use the stochastic PDE 2.17 (or 2.23) itself but its solution, Eq. 2.28. Taking the logarithm on both sides of this solution yields[1]:

$$\ln(S(t+\delta t)) = \ln(S(t)) + \mu\,\delta t + \sigma X\sqrt{\delta t} \quad \text{with} \quad X \sim \mathrm{N}(0,1)\ .$$

This is a *recursion*: Knowledge of $\ln(S)$ at time t enables the calculation of $\ln(S)$ at the next time point $t+\delta t$. To emphasize this point, we enumerate the time points, i.e., we introduce the following notation:

$$t_i = t + i\,\delta t \quad \text{where} \quad i = 0, 1, \ldots, n \quad \text{d. h.} \quad t_0 = t\ , \quad t_n = T\ .$$

Denoting the ith (standard normally distributed) random number by $X_i = X(t_i)$, we obtain the basis of the simulation of a risk factor whose behavior is governed by Eq. 2.17 in terms of the notation just introduced:

$$\ln(S(t_i)) = \ln(S(t_{i-1})) + \mu\,\delta t + \sigma\, X_i\sqrt{\delta t}\ , \quad i = 1, \ldots, n\ .. \tag{11.1}$$

This equation is suitable for a direct simulation: $\ln(S(t_{i-1}))$ is the end-point of a random walk after $i-1$ steps. In the next step, the value "$\mu\,\delta t + \sigma\, X_i\sqrt{\delta t}$" will be added on to this end point, yielding the end point of the random walk after i steps, namely $\ln(S(t_i))$.

A concrete example at this point may clarify this procedure. With a volatility of $\sigma = 20\%$ per year, a drift of $\mu = 6\%$ per year and a time step $\delta t = 1$ day $= 1/365$ years, Eq. 11.1 gives the following simple recursion relation:

$$\begin{aligned}\ln(S(t_i)) &= \ln(S(t_{i-1})) + \frac{0,06}{365} + 0,2\,\sqrt{1/356}\,X_i \\ &= \ln(S(t_{i-1})) + 0,0001644 + 0,01046\,X_i\ .\end{aligned}$$

The contribution of the drift to each step is approximately one hundred times smaller than that of the volatility. This explains the negligible effect of the drift for small time spans $T-t$. However, over longer time periods, the drift cannot be ignored. Figure 11.3 shows the result of a simulation over a longer

[1]To be able to work with standard normally distributed random numbers we also used Eqs. 2.16 and 2.27 here. Those Equations say that the Wiener-Process δW has the same distribution as $\sqrt{\delta t}$ times a standard normally distributed random number:

$$\delta W \sim X\sqrt{\delta t} \quad \text{with} \quad X \sim \mathrm{N}(0,1)\ .$$

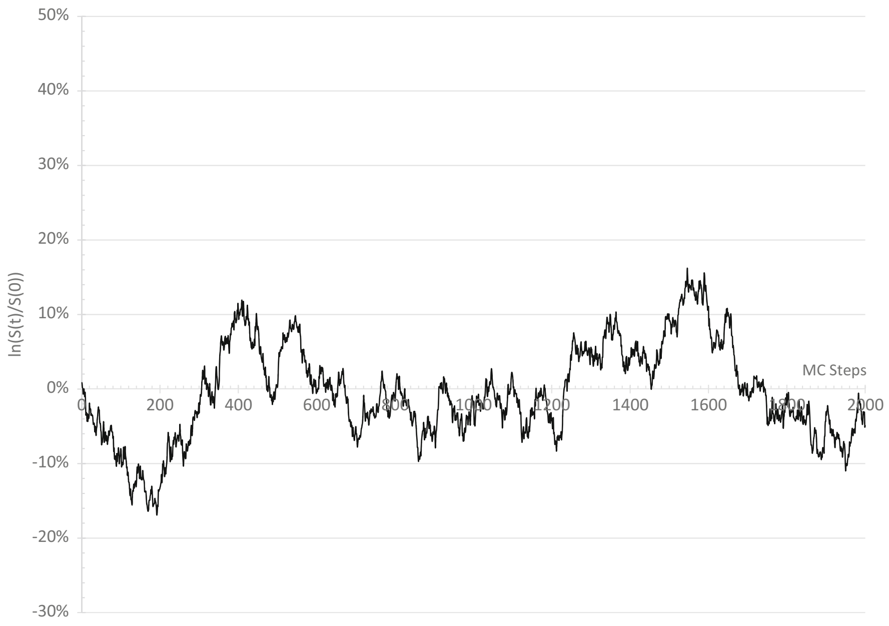

Fig. 11.2 Simulation of $\ln(S(t)/S(0))$ by means of normally distributed random numbers. 1 step corresponds to 1 day, with a total number of 2,000 steps. The annual volatility is 20% and the average drift is 0

time span of 2,000 days with respect to this recursion. To emphasize the effect of the drift, the 2,000 randomly generated values of X_i were saved and used again for the recursion, this time setting $\mu = 0$. The result is presented in Fig. 11.2. The values at the end of the simulation performed with the drift (Fig. 11.3) are approximately twice as large as those for the simulation with zero drift.

Such a curve represents one possible price progression over time, called a *path* of the risk factor. Repeating the simulation, we obtain an additional path. The simulation of many such paths yields the probability distributions for the price at each time point t_i in the simulated time span, in other words the probability distribution of the paths. In particular, we obtain a distribution of the values of the risk factor at the end point $t_n = T$. The simulated price of the risk factor at time $t_n = T$ is obtained by adding up the n steps generated in accordance with Eq. 11.1:

$$\ln(S(t_n)) = \ln(S(t_0)) + \mu \sum_{i=1}^{n} \delta t + \sum_{i=1}^{n} X_i \, \sigma \, \sqrt{\delta t} \; .$$

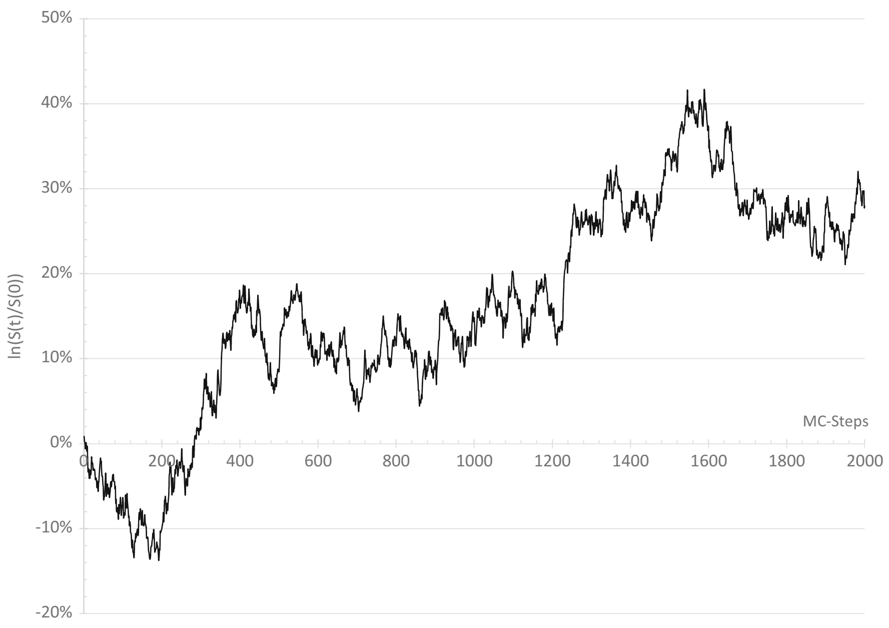

Fig. 11.3 The same random walk as in Fig. 11.2 but with a drift (mean return) of 6% per year

If the volatility σ is assumed to be constant, it can be factored out of the above sum. The length of the time steps δt are the same for each Monte Carlo step and thus

$$\ln(S(t_n)) = \ln(S(t_0)) + \mu\, n\, \delta t + \sigma \sqrt{\delta t} \sum_{i=1}^{n} X_i \ .$$

The sum of n independent, standard normally distributed random variables is again a normally distributed random variable with expectation 0 and variance n, i.e., the standard deviation of the sum is $\sqrt{n}$. If only the end distribution (the distribution of S_n) is of interest and not the *path* of the underlying (each of the S_i) taken *during* the time span under consideration, the sum of the n standard normally distributed random variables X_i can be replaced by a standard normally distributed random variable X multiplied by $\sqrt{n}$:

$$\sum_{i=1}^{n} X_i \rightarrow \sqrt{n}\, X \ .$$

In doing so, the simulated underlying price at the end of the time period under consideration can be generated with a *single* random number *directly*. Using $n\delta t = T - t$ from the definition of δt we can thus write

$$\ln(S(T)) = \ln(S(t)) + (T-t)\mu + \sigma\sqrt{T-t}\,X \,. \tag{11.2}$$

The path taken by the underlying *during* the simulation is of interest only for certain exotic, *path-dependent* derivatives (see Chap. 19). Otherwise, in particular for pricing and risk management of European-style instruments where the underlying price is relevant only at maturity (or at the end of the *liquidation period* in the case of risk management), the efficiency of a Monte Carlo simulation can be significantly improved by this simplification.

11.3.2 Simulation of *Several* Correlated Risk Factors

We are often interested in the progression of several risk factors rather than just one. The risk management of a portfolio, for example, requires the simulation of all risk factors affecting the portfolio. Since those are usually not statistically independent of one another, we are confronted with the question of how the *correlation* between risk factor processes can be incorporated into the simulation. The general approach for an arbitrarily large number of different securities is presented in Sect. 23.1. Here, we restrict the discussion to the important special case of *two* correlated prices. This can be used, for example, in determining the price of *exchange options*.[2]

The two price processes S_1 and S_2 have drifts μ_1 and μ_2, volatilities σ_1 and σ_2, respectively, and a correlation ρ_{12}. The logarithm of the random walks will again be used to model the time evolution of the risk factors:

$$\begin{aligned}
&\delta \ln S_1(t) = \mu_1\,\delta t \;+\; Y_1\\
&\delta \ln S_2(t) = \mu_2\,\delta t \;+ Y_2 \;\text{ with correlated random variables }\; Y_1,\; Y_2 \;.
\end{aligned}$$

How should correlated pairs of random variables be constructed? First, the two equations for $\delta\ln(S_i)$ are combined by interpreting the indexed equations as components of a random vector.

$$\begin{pmatrix}\delta\ \ln S_1(t)\\ \delta\ \ln S_2(t)\end{pmatrix} = \begin{pmatrix}\mu_1\\ \mu_2\end{pmatrix}\delta t + \begin{pmatrix}Y_1\\ Y_2\end{pmatrix} \;.$$

[2] See Sect. 19.1.5.

If the prices were uncorrelated, the random variables Y_i would be independent, normally distributed random variables with variance $\sigma_i^2 \delta t$. However, since the prices are correlated, it is not sufficient to simply specify the variance of both variables in order to fully determine the distribution. Instead the *co*variance is needed to describe both the variances *and* the correlations. Since several Y_i (in this case, two) may come into play when constructing the random vector, the covariance is not a single number but a matrix, called the *covariance matrix*. As will be presented in detail in Sect. 21.5, the covariance matrix $\delta\Sigma$ of two random variables is composed of the correlations and the standard deviations of the associated random variables as follows:

$$\delta\mathbf{\Sigma} = \begin{pmatrix} \delta\Sigma_{11} & \delta\Sigma_{12} \\ \delta\Sigma_{21} & \delta\Sigma_{22} \end{pmatrix} \text{ where } \delta\Sigma_{ij} = \underbrace{\rho_{ij}}_{\substack{\text{Correlation} \\ \text{von } i \text{ mit } j}} \quad \underbrace{\sigma_i \quad \sqrt{\delta t}}_{\substack{\text{Standard} \\ \text{dev. of} i}} \underbrace{\sigma_j \sqrt{\delta t}}_{\substack{\text{Standard} \\ \text{dev. of} j}} \text{for } i, j = 1, 2 \; . \tag{11.3}$$

Correlations are symmetric, i.e., $\rho_{ij} = \rho_{ji}$. Also, $\rho_{ii} = 1$, in other words a risk factor is always fully correlated with itself. With $\rho = \rho_{12} = \rho_{21}$ the covariance matrix of two risk factors becomes

$$\delta\mathbf{\Sigma} = \delta t \begin{pmatrix} \sigma_1^2 & \rho\,\sigma_1\sigma_2 \\ \rho\,\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} .$$

In order to generate normally distributed random numbers Y_i with correlation matrix $\delta\Sigma$ from independent, standard normally distributed random variables X_i, the "square root" $\mathbf{A}$ of the matrix $\delta\Sigma$ is needed. This matrix satisfies the condition

$$\mathbf{A}\mathbf{A}^T = \delta\Sigma \; ,$$

where $\mathbf{A}^T$ denotes the *transpose* of the matrix obtained by writing the column vectors of the matrix $\mathbf{A}$ as row vectors. As shown in detail in Sect. 21.5.3, this matrix yields the desired transformation

$$\begin{aligned} \mathbf{Y} &= \mathbf{AX} \quad \text{where} \\ \mathbf{X} &= \text{vector of standard normally distributed, uncorrelated random variables} \\ \mathbf{Y} &= \text{vector of normally distributed, correlated random variables with} \\ &\quad\ \text{covariance matrix } \delta\Sigma \; . \end{aligned}$$

The “square root of a matrix” can be obtained using a procedure from linear algebra called *Cholesky decomposition*. The general form of this decomposition is given in Sect. 21.5.3. Here, we restrict our consideration to 2×2 matrices, carrying out the procedure explicitly for this case. We begin by assuming that the matrix $\mathbf{A}$ has the following form:

$$\mathbf{A} = \begin{pmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{pmatrix} \Longrightarrow \mathbf{A}^T = \begin{pmatrix} a_{11} & a_{21} \\ 0 & a_{22} \end{pmatrix} .$$

The components a_{ij} can now be determined from the requirement that the equation $\mathbf{A}\mathbf{A}^T = \delta\boldsymbol{\Sigma}$ be satisfied:

$$\begin{gathered} \mathbf{A}\mathbf{A}^T = \delta\Sigma \\ \begin{pmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} a_{11} & a_{21} \\ 0 & a_{22} \end{pmatrix} = \begin{pmatrix} \delta\Sigma_{11} & \delta\Sigma_{12} \\ \delta\Sigma_{21} & \delta\Sigma_{22} \end{pmatrix} \\ \begin{pmatrix} a_{11}^2 & a_{11}a_{21} \\ a_{11}a_{21} & a_{21}^2 + a_{22}^2 \end{pmatrix} = \begin{pmatrix} \delta\Sigma_{11} & \delta\Sigma_{12} \\ \delta\Sigma_{21} & \delta\Sigma_{22} \end{pmatrix} . \end{gathered}$$

Comparing the components on both sides yields a linear system of equations for the a_{ij}:

$$\begin{aligned} a_{11}^2 = \delta\Sigma_{11} &\Rightarrow a_{11} = \sqrt{\delta\Sigma_{11}} = \sigma_1\sqrt{\delta t} \\ a_{11}a_{21} = \delta\Sigma_{12} &\Rightarrow a_{21} = \frac{\delta\Sigma_{12}}{\sqrt{\delta\Sigma_{11}}} = \rho\ \sigma_2\sqrt{\delta t} \\ a_{21}^2 + a_{22}^2 = \delta\Sigma_{22} &\Rightarrow a_{22} = \sqrt{\delta\Sigma_{22} - \frac{\delta\Sigma_{12}^2}{\delta\Sigma_{11}}} = \sqrt{1-\rho^2}\,\sigma_2\sqrt{\delta t} . \end{aligned}$$

Where Eq. 11.3 was also used. Now that the matrix elements have been determined, this matrix can be used to generate the correlated random numbers:

$$\begin{gathered} \mathbf{Y} = \mathbf{A}\mathbf{X} \\ \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \begin{pmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \sqrt{\delta t} \begin{pmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sqrt{(1-\rho^{2)}}\sigma_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} , \end{gathered}$$

and thus

$$
\begin{aligned}
Y_1 &= \sqrt{\delta t}\,\sigma_1\,X_1\\
Y_2 &= \sqrt{\delta t}\,\rho\,\sigma_2\,X_1 + \sqrt{\delta t}\,\sqrt{1-\rho^2}\,\sigma_2\,X_2\,.
\end{aligned}
\tag{11.4}
$$

We can now formulate a random walk equation for two *correlated* price processes in terms of two *un*correlated *standard* normally distributed random variables X_1, X_2:

$$
\begin{aligned}
\delta \ln S_1(t) &= \mu_1\,\delta t \;+\; \sigma_1\,X_1\sqrt{\delta t}\\
\delta \ln S_2(t) &= \mu_2\,\delta t \;+\; \sigma_2\,(\rho X_1 + \sqrt{1-\rho^2}X_2)\sqrt{\delta t}\,.
\end{aligned}
\tag{11.5}
$$

The first equation has the form of a single random walk. The correlation affects only the second equation. The interpretation of this representation is that the *second* price is correlated with the *first*. Since the correlations are symmetric, this interpretation is irrelevant for the final result. Assuming another form for the matrix **A** (or simply renaming the Y_i) would yield the result that the *first* price is correlated with the *second*. This holds in general: we can select any risk factor (for example, the one we feel most comfortable working with) as the leading factor and simulate it independently. Then all correlations appear in the evolutions of the other risk factors.

Exactly as in the previous section, the random walk equations 11.5 now provide the basis for the recursion which can be programmed in a Monte Carlo simulation:

$$
\begin{aligned}
\ln S_1(t_i) &= \ln S_1(t_{i-1}) + \mu_1\,\delta t \;+\; \sigma_1\,X_1(t_i)\sqrt{\delta t} \quad , \qquad i = 1,\,\dots\,,\,n\\
\ln S_2(t_i) &= \ln S_2(t_{i-1}) + \mu_2\,\delta t \;+\; \sigma_2\left[\rho X_1(t_i) + \sqrt{1-\rho^2}X_2(t_i)\right]\sqrt{\delta t}\,.
\end{aligned}
\tag{11.6}
$$

Or, if only the values at the end of the time period under consideration are of interest but not the paths of the risk factors:

$$
\begin{aligned}
\ln S_1(T) &= \ln S_1(t) + \mu_1\,(T-t) \;+\; \sigma_1\,\sqrt{T-t}X_1\\
\ln S_2(T) &= \ln S_2(t) + \mu_2\,(T-t) \;+\; \sigma_2\sqrt{T-t}\left[\rho X_1 + \sqrt{1-\rho^2}X_2\right]\,.
\end{aligned}
\tag{11.7}
$$

The Excel workbook from the download site [50] MonteCarloDemo.xlsx includes a demonstration of the Monte Carlo simulation of a risk factor. For demonstration purposes, the simulation is accomplished using Excel cell functions without programming in Visual-Basic. Of course, this would be too slow for use in real applications. We therefore also provide a workbook entitled MonteCarloSimulation.xlsm which contains an executable Visual Basic module for Monte Carlo simulations.

11.4 Pricing

According to Eq. 9.20 the value at time t of a financial instrument can be determined from the expectation of its price at a future time T, if this expectation is taken with respect to the risk-neutral probability distribution of the underlying. A clever choice of T can make pricing the instrument using Monte Carlo simulations quite easy. If T is chosen to be the maturity of the derivative, then the derivative's price at time T is simply given by the payoff profile P. Thus, if the underlying S is simulated according to Eq. 11.1 up to time T we can easily obtain a *simulated* probability distribution for the payoff values. This works also for path dependent instruments if we measure the relevant path-dependent quantities along the way (like for instance average of the simulated underlying values for specified days in the case of Asian options). The *mean* of all these simulated payoff values is then used as an estimator for the *expectation* of the payoff. According to Eq. 9.20, discounting[3] this estimator back to time t yields an estimator for the value of the instrument at time t, *if* the calculation of the mean has been carried out with respect to the risk-neutral probability. To ensure that this is the case, the risk factor must be *simulated* with respect to this probability. This is accomplished by simply choosing the *drift* of the random walk to be risk-neutral in accordance with Eq. 9.25.

Thus, the approach for pricing derivatives using Monte Carlo simulations is basically clear: We simulate the underlying (or more precisely the logarithm of the underlying value) in a risk-neutral world up to the time of maturity in accordance with Eq. 11.1 (for path-*in*dependent derivatives we can even use Eq. 11.2 and save a lot of computing time), and then measure the (discounted) mean of the payoff profile. In order to determine the error involved as discussed

[3]For *future styled* instruments, whose value changes are settled by daily adjustments in a margin account (as is the case for futures, for example), today's price is directly related to this expectation without any discounting, see Eqs. 6.6 and 9.21.

in Sect. 31.2, the mean of the square of the payoff profile must be measured, too. That's all. The calculation of option prices using the Monte Carlo method merely involves determining the mean of certain functions during the simulation.

An essential point, however, should not be overlooked. The time of exercise must be known, otherwise it is not clear "up to when" the price is to be simulated. For American or Bermudan Options we do not have this information up front. Section 11.5 describes methods for calculating at least lower bounds for the option values, which could be used to approximate the exact solution quite accurately.

A detailed demonstration of the application of the Monte Carlo method to the valuation of an option portfolio is provided in the Excel workbook MonteCarloSimulation.xlsx from the download site [50]. By making appropriate adjustments in the valuation part of the Visual Basic module, this program can be used to price all sorts of European derivatives[4] in the context of the Black-Scholes world (constant yields and volatility). In anticipation of Chap. 12, the calculation of *Greeks*, i.e., the sensitivities of the derivative's price with respect to its parameters, is also demonstrated in the workbook. The workbook can be used as a small but complete option calculator (as always, the yellow fields are the input fields).

11.5 American Monte Carlo

The Monte Carlo methods can be extended for the valuation for options with a Bermudan or American exercise profile. As first step, American exercise can be approximated by Bermudan exercise. The higher the frequency of Bermudan exercise days (i.e. daily), the better the continuous exercise right will be approximated. For numerical computations, this discretization of continuous time to a set of distinct time points is anyway required. Therefore, it is only a matter of required accuracy and available computer time, how small the time steps must resp. should be chosen. Thus, in the following, we only consider Bermudan exercise rights.

The second problem is more difficult to solve. At each potential exercise day t_e we need to decide, whether the option should be exercised or not. Besides the price of the underlying (directly simulated by the Monte Carlo method) used to calculate the pay off in case of exercise (exercise value), we need to

[4]See also Sect. 19.3.1.

compute the price of the option to calculate the value of exercise at some later point of time (hold value). A simple approach would start for simulated path at each exercise date a new Monte Carlo simulation with many paths to simulate the future price of the option. With this approach, computing time increases dramatically with the number of exercise dates of the option and the number of simulated paths. Similar to non-recombining tree, this method is usually not feasible. Therefore, various authors have suggested alternative methods, of which the most widely known (and used) is probably the work of Longstaff ans Schwartz [135]. This method is also known as the Longstaff Schwartz method. Another, alternative approach, will be presented in Sect. 14.10.

The general idea of American Monte Carlo method is to calculate the optimal exercise time based on key figures well known at the time of exercise. This key figures should have the best explanatory power for the considered problem. For a Plain Vanilla American option, this could be the relative moneyness

$$x = \frac{K - S(t_e)}{S(t_e)} \ .$$

Here, $S(t_e)$ is the simulated underlying price at the (potential) exercise time. K is, as usual, the strike of the option. Then, the hold value can be modeled as a function $h(x)$. E.g., a simple approach would be a 2nd degree polynomial.

$$h(x) = a_2 x^2 + a_1 x + a_0 \ .$$

In the beginning, the 3 parameters a_0, a_1, a_2 are unknown and need to be determined. This is down in a simulation run preceding the final simulation for calculating the full option. In this pre-run, only the underlying price will be simulated. For each path $i = 1, 2, \ldots, n$ and each possible exercise date t_j with $t > t_j < T$ and $j = 1, 2, \ldots, m$, the simulated underlying price $S_{i,j}$ is computed and stored. Then, we will go backwards in time through all paths. At the final exercise date t_m, the hold value of the option can be computed (it will be either exercised at this date or expires worthless) as pay off at time T discounted from T to t_m. For discounting, the numeraire simulated for each path can be used. For each of the n paths, the hold value H_m is calculated separately. Then, using an optimization algorithm, the 3 parameters a_0, a_1, a_2 will be determined such that the n values H_m can be approximated by the function $h(x)$ as close as possible. By approximating H_m by $h(x)$, we describe the hold value by means of parameter x which is well known at time t_m without using the information of the future (unknown) underlying price at time T we needed to calculate H_m.

Next, we can go one step back to date t_{m-1} and calculate the hold values H_{m-1} for each path. Now, we need take into account the possibility of exercise at time t_m. This would be exactly the case if the inner value $c(t_m)$ of the option at date t_m is worth more than the hold value, since the option buyer seeks to maximize the profit. Therefore, for each path, the hold value H_{m-1} is calculated as discounted value of $\max[c(t_m), H_m]$. After the hold values have been determined, we use again an optimizer over all paths to determine another set of three parameters a_0, a_1, a_2 to minimize the differences between $h(x)$ and the H_{m-1}. This procedure is repeated until for each exercise date an optimal set of parameters has been found. Finally, we have n functions $h_i(x)$ determined, each function modeling the hold value of the option at one exercise date.

At last, the simulation to calculate the full Bermudan option could be exercised. For this, we simulate again many Monte Carlo paths (in general more than has been used for optimizing the parameters for approximating the hold value, e.g. $10n$ paths). As with the "normal" Monte Carlo method, we will go forward in time. At each exercise date t_j we check, whether $c(t_j) > h_j(x)$ is fulfilled. If so, the option will be exercised. If not, we continue the simulation to the next exercise date, until either the option is exercised or matured. Then, the price can be calculated the sum of all cash flows per path discounted with the numeraire averaged over all simulated paths. To avoid a numerical bias, it is important to make sure that the random numbers used in the pre-run and those in the second (full) run are independent.

In this way, it is possible to compute a lower bound to the actual value for the Bermudan option. The better the choice of function $h(\cdot)$, the better the lower bound will approximate the real value. This can be seen very easily: In the end, $h(\cdot)$ serves as a mathematically fixed exercise strategy and is only dependent on parameters well known at the exercise date. If we have made a bad choice for $h(\cdot)$, our exercise decision will be suboptimal and we would not be able to receive the full value of the option.

A drawback of this approach is the fact that a good choice for $h(\cdot)$ is often not easily found. The choice of moneyness as explanatory variable in a second order polynomial is probably not the best choice, if the well-known functional form of the intrinsic value of a Amerikan Plain Vanilla option is envisaged. The weighted sum of European Plain Vanilla options for each exercise date might be a better choice. Unless there is no benchmark price, a polynomial is a good starting point. Since we know that we will get only a lower limit, it is useful to experiment with various approaches for $h(\cdot)$ and finally chose that function, for which the simulated American options had maximum value.