Chapter 3 Monte Carlo Simulation with Stochastic Differential Equations

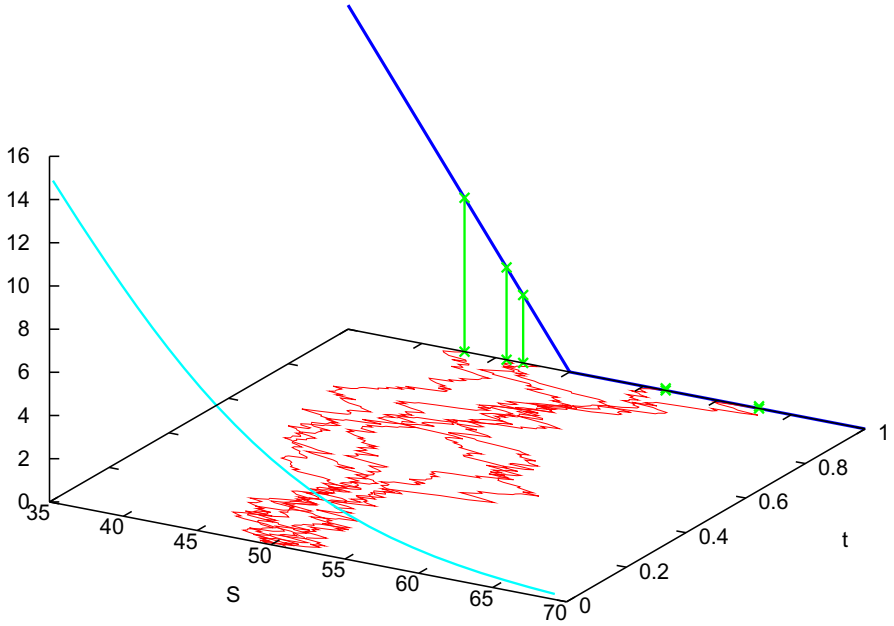

Fig. 3.1. Illustration of the Monte Carlo approach for a European put, with $K = 50$, $S_0 = 50$, $T = 1$, $\sigma = 0.2$, $r = 0$; five simulated paths in the (S, t)-plane with payoff; vertical axis: V. The front curve $V(S, 0)$ is shown.

The Sections 1.5 and 1.7.3 have introduced the principle of risk-neutral evaluation, which can be summarized by

$$V(S_0, 0) = \mathrm{e}^{-rT} \mathsf{E}_{\mathsf{Q}}(V(S_T, T) \mid S_t \text{ starting from } (S_0, 0)),$$

where E_{Q} represents the expectation under a risk-neutral measure. For the Black–Scholes model, this expectation is an integral as in (1.50). This suggests two approaches of calculating V. Either approximate the integral, or calculate the expectation by simulating the underlying stochastic differential equation (SDE) repeatedly. The latter approach is illustrated in Figure 3.1. Five paths S_t are calculated for $0 \le t \le T$ in the risk-neutral fashion, each starting from S_0. Then for each resulting S_T the payoff is calculated, here for a European put. The figure illustrates the bulk of the work. (In reality, thousands of paths are calculated.) It remains the comparably cheap task of calculating the mean of the payoffs as approximation for E_{Q}. This is the Monte Carlo approach. The Monte Carlo approach works for general models, for example, for systems of equations, see Figure 3.2.

R.U. Seydel, *Tools for Computational Finance*, Universitext,
DOI 10.1007/978-1-4471-2993-6_3, © Springer-Verlag London Limited 2012

This chapter is based on the ability to numerically integrate SDEs. Therefore a significant part of the chapter is devoted to this topic. Again X_t denotes a stochastic process and a solution of an SDE (1.31),

$$\mathrm{d}X_t = a(X_t, t)\,\mathrm{d}t + b(X_t, t)\,\mathrm{d}W_t \quad \text{for } 0 \le t \le T\,,$$

where the driving process W is a Wiener process. We assume a t-grid with $0 = t_0 < t_1 < \ldots$. For convenience, the step length $\Delta t = t_{j+1} - t_j$ is taken equidistant. As is common usage in numerical analysis, we also use the h-notation, $h := \Delta t$. For $\Delta t = h = T/m$ the index j runs from 0 to $m-1$. The solution of a discrete version of the SDE is denoted y_j. That is, y_j should be an approximation to X_{t_j}, or y_t an approximation to X_t. Weaker requirements will be discussed below. The initial value for $t = 0$ is assumed a given constant,

$$y_0 = X_0\,.$$

For example, from Algorithm 1.11 we know the Euler discretization

$$\begin{cases} y_{j+1} = y_j + a(y_j, t_j)\Delta t + b(y_j, t_j)\Delta W_j\,, \quad t_j = j\Delta t\,, \\ \Delta W_j = W_{t_{j+1}} - W_{t_j} = Z\sqrt{\Delta t} \quad \text{with } Z \sim \mathcal{N}(0,1)\,. \end{cases} \tag{3.1}$$

Since an approximation y_T also depends on the chosen step length h, we also write y_T^h. From numerical methods for deterministic ODEs ($b \equiv 0$) we know the discretization error of Euler's method is $O(h)$,

$$X_T - y_T^h = O(h)\,.$$

The Algorithm 1.11 (repeated in equation (3.1)) is an *explicit* method in that in every step $j \to j+1$ the values of the functions a and b are evaluated at the previous approximation (y_j, t_j). Evaluating b at the left-hand mesh point (y_j, t_j) is consistent with the Itô integral and the Itô process, compare the notes at the end of Chapter 1.

After we have seen in Chapter 2 how $Z \sim \mathcal{N}(0,1)$ can be calculated, all elements of Algorithm 1.11 are known, and we are equipped with a method to numerically integrate SDEs ($\longrightarrow$ Exercise 3.1). In this chapter we learn about other methods, and discuss the accuracy of numerical solutions of SDEs. The exposition of Sections 3.1 through 3.3 follows [KlP92]. Readers content with Euler's method (3.1) may like to skip these sections.

After a brief exposition on constructing bridges (Section 3.4), we turn to the main theme, namely, Monte Carlo methods for pricing options. The basic principle is outlined for European options (Section 3.5). For American options parametric methods and regression methods are introduced in Section 3.6. The final Section 3.7 discusses the calculation of sensitivities.

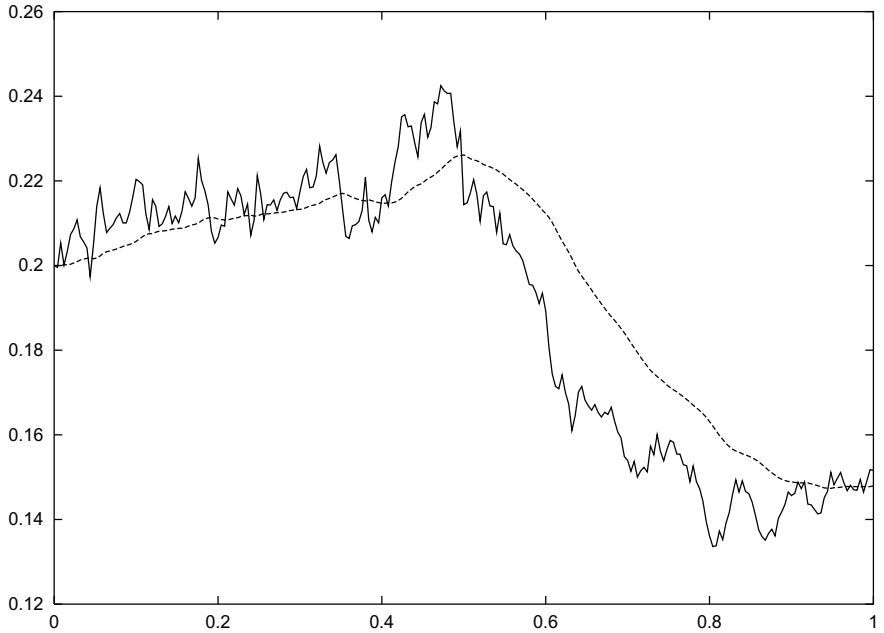

Fig. 3.2. Example 1.15, $\alpha = 0.3$, $\beta = 10$, $\sigma_0 = \zeta_0 = 0.2$, realization of the volatility tandem σ_t, ζ_t (dashed) for $0 \le t \le 1$, $\Delta t = 0.004$

3.1 Approximation Error

To study the accuracy of numerical approximations, we choose the example of a linear SDE

$$\mathrm{d}X_t = \alpha X_t \,\mathrm{d}t + \beta X_t \,\mathrm{d}W_t, \qquad \text{initial value } X_0 \text{ for } t = 0\,.$$

For this equation with constant coefficients α, β we derived in Section 1.8 the analytical solution

$$X_t = X_0 \exp\left(\left(\alpha - \tfrac{1}{2}\beta^2\right) t + \beta W_t\right)\,. \tag{3.2}$$

For a given realization of the Wiener process W_t we obtain as solution a trajectory *(sample path)* X_t. For another realization of the Wiener process the same theoretical solution (3.2) takes other values. If a Wiener process W_t is given, we call a solution X_t of the SDE a *strong solution*. In this sense the solution (3.2) is a strong solution. If one is free to select a Wiener process, then a solution of the SDE is called *weak solution*. For a weak solution, only the distribution of X is of interest, not its path.

Assuming an identical sample path of a Wiener process for the SDE and for a numerical approximation y_t^h, a pathwise comparison of the trajectories X_t with y_t^h is possible for all t_j. For example, for $t_m = T$ the absolute error

of a strong solution for a given Wiener process is $|X_T - y_T^h|$. For another path of the Wiener process the error is somewhat different. We average the error over "all" sample paths of the Wiener process:

Definition 3.1 (absolute error)
For a strong solution X_t of the SDE with approximation y_t^h the absolute error at T is $\epsilon(h) := \mathsf{E}(|X_T - y_T^h|)$.

In practice we represent the set of all sample paths of a Wiener process by N different simulations.

Example 3.2 (Euler method)
For the SDE with $X_0 = 50$, $\alpha = 0.06$, $\beta = 0.3$, $T = 1$ we investigate experimentally how the absolute error of the Euler method (3.1) depends on h. Starting with a first choice of h we calculate $N = 50$ simulations and for each realization the values of X_T and y_T —that is $X_{T,k}$, $y_{T,k}$ for $k = 1, ..., N$. Again: to obtain pairs of comparable trajectories, also the theoretical solution (3.2) is fed with the same Wiener process used in (3.1). Then we calculate the estimate $\widehat{\epsilon}$ of the absolute error ϵ,

$$\widehat{\epsilon}(h) := \frac{1}{N} \sum_{k=1}^{N} |X_{T,k} - y_{T,k}^h| \,.$$

Such an experiment was performed for five values of h. In this way the first series of results were obtained (first line in Table 3.1). Such a series of experiments was repeated twice, using other seeds. As Table 3.1 shows, $\widehat{\epsilon}(h)$ decreases with decreasing h, but slower than one would expect from the behavior of the Euler method applied to deterministic differential equations. The order can be determined by fitting the values of the table. We bypass this and test the order $O(h^{1/2})$ right away. To this end, divide each $\widehat{\epsilon}(h)$ of the table by the corresponding $h^{1/2}$. This shows that the order $O(h^{1/2})$ is correct, because each entry of the table leads essentially to the same constant value, here 2.8. Apparently this example satisfies $\widehat{\epsilon}(h) \approx 2.8\, h^{1/2}$. For another example we would expect a different constant.

Table 3.1. Results of Example 3.2

Table of the $\widehat{\epsilon}(h)$	$h = 0.01$	$h = 0.005$	$h = 0.002$	$h = 0.001$	$h = 0.0005$
series 1 (with seed$_1$)	0.2825	0.183	0.143	0.089	0.070
series 2 (with seed$_2$)	0.2618	0.195	0.126	0.069	0.062
series 3 (with seed$_3$)	0.2835	0.176	0.116	0.096	0.065

These results obtained for the estimates $\widehat{\epsilon}$ are assumed to be valid for ϵ. This leads to postulate

$$\epsilon(h) \le c\, h^{1/2} = O(h^{1/2}).$$

The order of convergence is worse than the order $O(h)$, which Euler's method (3.1) achieves for deterministic differential equations ($b \equiv 0$). But in view of (1.28), $(\mathrm{d}W)^2 = h$, the order $O(h^{1/2})$ is no surprise. For a proof of the order, see [KlP92].

Definition 3.3 (strong convergence)
y_T^h converges strongly to X_T with order $\gamma > 0$,
if $\epsilon(h) = \mathsf{E}(|X_T - y_T^h|) = O(h^\gamma)$.
y_T^h converges strongly, if

$$\lim_{h \to 0} \mathsf{E}(|X_T - y_T^h|) = 0\,.$$

Hence the Euler method applied to SDEs converges strongly with order 1/2. Note that convergence refers to fixed finite intervals, here for a fixed value T. For long-time integration ($T \to \infty$), see the Notes at the end of this chapter.

Strongly convergent methods are appropriate when the trajectory itself is of interest. This was the case for Figures 1.16 and 1.17. Often the pointwise approximation of X_t is not our real aim but only an intermediate result in the effort to calculate a **moment**. For example, many applications in finance need to approximate $\mathsf{E}(X_T)$. A first conclusion from this situation is that of all calculated y_i only the last is required, namely, y_T. A second conclusion is that for the expectation a single sample value of y_T is of little interest. The same holds true if the ultimate interest is $\mathsf{Var}(X_T)$ rather than X_T. In this situation the primary interest is not strong convergence with the demanding requirement $y_T \approx X_T$ and even less $y_t \approx X_t$ for $t < T$. Instead the concern is the weaker requirement to approximate moments or other functionals of X_T. Then the aim is to achieve $\mathsf{E}(y_T) \approx \mathsf{E}(X_T)$, or $\mathsf{E}(|y_T|^q) \approx \mathsf{E}(|X_T|^q)$, or more general $\mathsf{E}(g(y_T)) \approx \mathsf{E}(g(X_T))$ for an appropriate function g.

Definition 3.4 (weak convergence)
y_T^h converges weakly to X_T with respect to g with order $\beta > 0$,
if $\mathsf{E}(g(X_T)) - \mathsf{E}(g(y_T^h)) = O(h^\beta)$.
y_T^h converges weakly to X_T with order β,
if this holds for all polynomials g.

The Euler scheme is weakly $O(h^1)$ convergent provided the coefficient functions a and b are four times continuously differentiable ([KlP92], Chapter 14). For the special polynomial $g(x) = x$, (B1.4) implies convergence of the mean $\mathsf{E}(x)$. For $g(x) = x^2$ the relation $\mathsf{Var}(X) = \mathsf{E}(X^2) - (\mathsf{E}(X))^2$ implies convergence of the variance (the reader may check). Proceeding in this way implies weak convergence with respect to all moments.

Since the properties of the integrals on which expectation is based lead to

$$|\mathsf{E}(X) - \mathsf{E}(Y)| = |\mathsf{E}(X - Y)| \leq \mathsf{E}(|X - Y|)\,,$$

we confirm that strong convergence implies weak convergence with respect to $g(x) = x$.

When weakly convergent methods are evaluated, the outcomes X_T and y_T need not be based on the same stochastic process, only their probability distributions must be close. This allows for a simplification of Euler's method. The increments ΔW can be replaced by other random variables $\Delta\widehat{W}$ that have the same expectation and variance. ΔW_j can be replaced by the simple approximation $\Delta\widehat{W}_j = \pm\sqrt{\Delta t}$, where each sign occurs with probability 1/2. The moments match; in particular, expectation and variance of $\Delta\widehat{W}$ and ΔW are the same: $\mathsf{E}(\Delta\widehat{W}) = 0$, $\mathsf{E}(\Delta\widehat{W}^2) = \Delta t$. The replacing random variables $\Delta\widehat{W}_j$ are by far easier to evaluate, costs can be saved significantly (⟶ Exercise 3.15). The simplified Euler method is again weakly convergent with order 1.

3.2 Stochastic Taylor Expansion

The derivation of algorithms for the integration of SDEs is based on stochastic Taylor expansions. To facilitate the understanding of stochastic Taylor expansions we confine ourselves to the scalar and autonomous[1] case, and first introduce the terminology by means of the deterministic case. That is, we begin with $\frac{\mathrm{d}}{\mathrm{d}t}X_t = a(X_t)$. The chain rule for arbitrary $f \in \mathcal{C}^1(\mathbb{R})$ is

$$\frac{\mathrm{d}}{\mathrm{d}t}f(X_t) = a(X_t)\frac{\partial}{\partial x}f(X_t) =: Lf(X_t)\,.$$

With the linear operator L this rule in integral form is

$$f(X_t) = f(X_{t_0}) + \int_{t_0}^{t} Lf(X_s)\,\mathrm{d}s\,. \tag{3.3}$$

This version is resubstituted for the integrand $\tilde{f}(X_s) := Lf(X_s)$, which requires at least $f \in \mathcal{C}^2$, and gives the term in braces:

$$f(X_t) = f(X_{t_0}) + \int_{t_0}^{t} \left\{ \tilde{f}(X_{t_0}) + \int_{t_0}^{s} L\tilde{f}(X_z)\,\mathrm{d}z \right\}$$

[1] An *autonomous* differential equation does not explicitly depend on the independent variable, here $a(X_t)$ rather than $a(X_t, t)$. The standard GBM Model 1.13 of the stock market is autonomous for constant μ and σ.

$$
\begin{aligned}
&= f(X_{t_0}) + \tilde{f}(X_{t_0}) \int_{t_0}^{t} \mathrm{d}s + \int_{t_0}^{t} \int_{t_0}^{s} L\tilde{f}(X_z)\, \mathrm{d}z\, \mathrm{d}s \\
&= f(X_{t_0}) + Lf(X_{t_0})(t - t_0) + \int_{t_0}^{t} \int_{t_0}^{s} L^2 f(X_z)\, \mathrm{d}z\, \mathrm{d}s
\end{aligned}
$$

This version of the Taylor expansion consists of two terms and the remainder as double integral. To get the next term of the second-order derivative, apply (3.3) for $L^2 f(X_z)$, and split off the term

$$
L^2 f(X_{t_0}) \int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}z\, \mathrm{d}s = L^2 f(X_{t_0}) \frac{1}{2} (t - t_0)^2
$$

from the remainder double integral. At this stage, the remainder is a triple integral. This procedure is repeated to obtain the Taylor formula in integral form. Each further step requires more differentiability of f.

We now devote our attention to stochastic diffusion and investigate the *Itô-Taylor expansion* of the autonomous scalar SDE

$$
\mathrm{d}X_t = a(X_t)\, \mathrm{d}t + b(X_t)\, \mathrm{d}W_t \,.
$$

Itô's Lemma for $g(x,t) := f(x)$ is

$$
\mathrm{d}f(X_t) = \{ \underbrace{a \frac{\partial}{\partial x} f(X_t) + \frac{1}{2} b^2 \frac{\partial^2}{\partial x^2} f(X_t)}_{=: L^0 f(X_t)} \}\, \mathrm{d}t + \underbrace{b \frac{\partial}{\partial x} f(X_t)}_{=: L^1 f(X_t)}\, \mathrm{d}W_t \,,
$$

or in integral form

$$
\boxed{f(X_t) = f(X_{t_0}) + \int_{t_0}^{t} L^0 f(X_s)\, \mathrm{d}s + \int_{t_0}^{t} L^1 f(X_s)\, \mathrm{d}W_s \,.} \tag{3.4}
$$

This SDE will be applied for different choices of f. Specifically for $f(x) \equiv x$ the SDE (3.4) recovers the original SDE

$$
X_t = X_{t_0} + \int_{t_0}^{t} a(X_s)\, \mathrm{d}s + \int_{t_0}^{t} b(X_s)\, \mathrm{d}W_s \,. \tag{3.5}
$$

First apply (3.4) to $f = a$ and to $f = b$. The resulting versions of (3.4) are substituted in (3.5) leading to

$$
\begin{aligned}
X_t = &X_{t_0} + \int_{t_0}^{t} \left\{ a(X_{t_0}) + \int_{t_0}^{s} L^0 a(X_z)\, \mathrm{d}z + \int_{t_0}^{s} L^1 a(X_z)\, \mathrm{d}W_z \right\} \mathrm{d}s \\
&+ \int_{t_0}^{t} \left\{ b(X_{t_0}) + \int_{t_0}^{s} L^0 b(X_z)\, \mathrm{d}z + \int_{t_0}^{s} L^1 b(X_z)\, \mathrm{d}W_z \right\} \mathrm{d}W_s
\end{aligned}
$$

with

$$\begin{aligned} L^0 a &= aa' + \tfrac{1}{2}b^2 a'' \qquad L^0 b = ab' + \tfrac{1}{2}b^2 b'' \\ L^1 a &= ba' \qquad\qquad\qquad L^1 b = bb' \,. \end{aligned} \tag{3.6}$$

Summarizing the four double integrals into one remainder expression R, we have

$$X_t = X_{t_0} + a(X_{t_0}) \int_{t_0}^{t} \mathrm{d}s + b(X_{t_0}) \int_{t_0}^{t} \mathrm{d}W_s + R \,, \tag{3.7a}$$

with

$$\begin{aligned} R = &\int_{t_0}^{t} \int_{t_0}^{s} L^0 a(X_z)\, \mathrm{d}z\, \mathrm{d}s + \int_{t_0}^{t} \int_{t_0}^{s} L^1 a(X_z)\, \mathrm{d}W_z\, \mathrm{d}s \\ &+ \int_{t_0}^{t} \int_{t_0}^{s} L^0 b(X_z)\, \mathrm{d}z\, \mathrm{d}W_s + \int_{t_0}^{t} \int_{t_0}^{s} L^1 b(X_z)\, \mathrm{d}W_z\, \mathrm{d}W_s \,. \end{aligned} \tag{3.7b}$$

The order of the terms is limited by the number of repeated integrations. In view of (1.28), $\mathrm{d}W^2 = \mathrm{d}t$, we expect the last of the integrals in (3.7b) to be of first order (and show this below).

In an analogous fashion the integrands in (3.7b) can be replaced using (3.4) with appropriately chosen f. In this way triple integrals occur. We illustrate this for the integral on $f = L^1 b$, which is the double integral of lowest order. The non-integral term of (3.4) allows to split off another "ground integral" with constant integrand,

$$R = L^1 b(X_{t_0}) \int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}W_z\, \mathrm{d}W_s + \tilde{R} \,.$$

In view of (3.6) and (3.7a) this result can be summarized as

$$\begin{aligned} X_t = &X_{t_0} + a(X_{t_0}) \int_{t_0}^{t} \mathrm{d}s + b(X_{t_0}) \int_{t_0}^{t} \mathrm{d}W_s \\ &+ b(X_{t_0}) b'(X_{t_0}) \int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}W_z\, \mathrm{d}W_s + \tilde{R} \,. \end{aligned} \tag{3.8}$$

A general treatment of the Itô-Taylor expansion with an appropriate formalism is found in [KlP92].

The next step is to formulate numerical algorithms out of the equations derived by the stochastic Taylor expansion. To this end the integrals must be solved. For (3.8) we need a solution of the double integral. For $X_t = W_t$ the Itô Lemma with $a = 0$, $b = 1$ and $y = g(x) := x^2$ leads to the equation $\mathrm{d}(W_t^2) = \mathrm{d}t + 2W_t\, \mathrm{d}W_t$. Specifically for $t_0 = 0$ this is the equation

$$\int_0^t \int_0^s \mathrm{d}W_z\, \mathrm{d}W_s = \int_0^t W_s\, \mathrm{d}W_s = \tfrac{1}{2}W_t^2 - \tfrac{1}{2}t \,. \tag{3.9}$$

Another derivation of (3.9) uses

$$\sum_{j=1}^{n} W_{t_j}(W_{t_{j+1}} - W_{t_j}) = \tfrac{1}{2}W_t^2 - \tfrac{1}{2}\sum_{j=1}^{n}(W_{t_{j+1}} - W_{t_j})^2$$

for $t = t_{n+1}$ and $t_1 = 0$, and takes the limit in the mean on both sides ($\longrightarrow$ Exercise 3.2). The general version of (3.9) needed for (3.8) is

$$\int_{t_0}^{t} W_s \,\mathrm{d}W_s = \tfrac{1}{2}\,(W_t - W_{t_0})^2 - \tfrac{1}{2}(t - t_0)\,.$$

With $\Delta t := t - t_0$ and the random variable $\Delta W_t := W_t - W_{t_0}$ this is rewritten as

$$\int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}W_z \,\mathrm{d}W_s = \tfrac{1}{2}\,(\Delta W_t)^2 - \tfrac{1}{2}\Delta t\,. \tag{3.10}$$

Since this double integral is of order Δt, it completes the list of first-order terms.

Also the three other double integrals

$$\int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}z \,\mathrm{d}s\ , \quad \int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}W_z \,\mathrm{d}s\ , \quad \int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}z \,\mathrm{d}W_s$$

are needed for the construction of higher-order numerical methods. The first integral is elementary, it is of second order and not stochastic. The two others depend on each other via the equation

$$\int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}z \,\mathrm{d}W_s + \int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}W_z \,\mathrm{d}s = \int_{t_0}^{t} \mathrm{d}W_s \int_{t_0}^{t} \mathrm{d}s \tag{3.11}$$

($\longrightarrow$ Exercise 3.3). This indicates that the two remaining double integrals are of order $(\Delta t)^{3/2}$. We will return to these integrals in the following section.

3.3 Examples of Numerical Methods

Now we apply the stochastic Taylor expansion to construct numerical methods for SDEs. First we check how Euler's method (3.1) evolves. Here we evaluate the integrals in (3.7a) and substitute

$$t_0 \to t_j, \quad t \to t_{j+1} = t_j + \Delta t\,.$$

This leads to

$$X_{t_{j+1}} = X_{t_j} + a(X_{t_j})\Delta t + b(X_{t_j})\Delta W_j + R\,.$$

After neglecting the remainder R the Euler scheme of (3.1) results, here for autonomous SDEs.

To obtain higher-order methods, further terms of the stochastic Taylor expansions are added. We may expect a "repair" of the half-order $O(\sqrt{\Delta t})$ by including the lowest-order double integral of (3.8), which is calculated in (3.10). The resulting correction term, after multiplying with bb', is added to the Euler scheme. Discarding the remainder $\widetilde{R}$, an algorithm results, which is due to [Mil74].

Algorithm 3.5 (Milstein)

Start: $t_0 = 0,\ y_0 = X_0,\ W_0 = 0,\ \Delta t = T/m$

loop $j = 0, 1, 2, ..., m-1:$

$t_{j+1} = t_j + \Delta t$

Calculate the values $a(y_j),\ b(y_j),\ b'(y_j)$

$\Delta W = Z\sqrt{\Delta t}$ with $Z \sim \mathcal{N}(0,1)$

$$y_{j+1} = y_j + a\Delta t + b\Delta W + \frac{1}{2}bb' \cdot ((\Delta W)^2 - \Delta t)$$

This integration method by Milstein is strongly convergent with order one (⟶ Exercise 3.8). Adding the correction term has raised the strong convergence order of Euler's method to 1.

Runge–Kutta Methods

A disadvantage of the Taylor-expansion methods is the use of the derivatives $a',\ b',\ ...$ Analogously as with deterministic differential equations there is the alternative of Runge–Kutta–type methods, which only evaluate a or b for appropriate arguments.

As an example we discuss the factor bb' of Algorithm 3.5, and see how to replace it by an approximation. Starting from

$$b(y + \Delta y) - b(y) = b'(y)\Delta y + O((\Delta y)^2)$$

and using $\Delta y = a\Delta t + b\Delta W$ we deduce in view of (1.28) that

$$\begin{aligned} b(y + \Delta y) - b(y) &= b'(y)(a\Delta t + b\Delta W) + O(\Delta t) \\ &= b'(y)b(y)\Delta W + O(\Delta t)\,. \end{aligned}$$

Applying (1.28) again, we substitute $\Delta W = \sqrt{\Delta t}$ and arrive at an $O(\sqrt{\Delta t})$-approximation of the product bb', namely,

$$\frac{1}{\sqrt{\Delta t}}\left(b[y_j + a(y_j)\Delta t + b(y_j)\sqrt{\Delta t}] - b(y_j)\right)\,.$$

This expression is used in the Milstein scheme of Algorithm 3.5. The resulting variant

$$\begin{aligned} \widehat{y} :=& y_j + a\Delta t + b\sqrt{\Delta t} \\ y_{j+1} =& y_j + a\Delta t + b\Delta W + \frac{1}{2\sqrt{\Delta t}}(\Delta W^2 - \Delta t)[b(\widehat{y}) - b(y_j)] \end{aligned} \tag{3.12}$$

is a Runge–Kutta method, which also converges strongly with order one. Versions of these schemes for nonautonomous SDEs read analogously.

Taylor Scheme with Weak Second-Order Convergence.

Next we investigate the method that results when in the remainder term (3.7b) the ground integrals of all double integrals are split off. This is done by applying (3.4) for $f = L^0a,\ f = L^1a,\ f = L^0b,\ f = L^1b$. Then the new remainder $\tilde{R}$ consists of triple integrals. For $f = L^1b$ this analysis was carried out at the end of Section 3.2. With (3.6) and (3.10) the correction term

$$bb'\frac{1}{2}\left((\Delta W)^2 - \Delta t\right)$$

has resulted, leading to the strong convergence order one of the Milstein scheme. For $f = L^0a$ the integral is not stochastic and the term

$$\left(aa' + \frac{1}{2}b^2a''\right)\frac{1}{2}\Delta t^2$$

is an immediate consequence. For $f = L^1a$ and $f = L^0b$ the integrals are again stochastic, namely,

$$I_{(1,0)} := \int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}W_z\ \mathrm{d}s = \int_{t_0}^{t}(W_s - W_{t_0})\ \mathrm{d}s\,,$$
$$I_{(0,1)} := \int_{t_0}^{t}\int_{t_0}^{s} \mathrm{d}z\ \mathrm{d}W_s = \int_{t_0}^{t}(s - t_0)\ \mathrm{d}W_s\,.$$

Summarizing all terms, the preliminary numerical scheme is

$$\begin{aligned} y_{j+1} = y_j + a\Delta t + b\Delta W + \frac{1}{2}bb'\left((\Delta W)^2 - \Delta t\right) \\ + \frac{1}{2}\left(aa' + \frac{1}{2}b^2a''\right)\Delta t^2 + ba'I_{(1,0)} + \left(ab' + \frac{1}{2}b^2b''\right)I_{(0,1)}\,. \end{aligned} \tag{3.13}$$

It remains to approximate the two stochastic integrals $I_{(0,1)}$ and $I_{(1,0)}$. Setting $\Delta Y := I_{(1,0)}$ we have in view of (3.11)

$$I_{(0,1)} = \Delta W\Delta t - \Delta Y\,.$$

At this state the two stochastic double integrals $I_{(0,1)}$ and $I_{(1,0)}$ are expressed in terms of only one random variable ΔY, in addition to the variable ΔW used before. Since for weak convergence only the correct moments are needed, all occurring random variables (here ΔW and ΔY) can be replaced by other random variables with the same moments. The normally distributed random variable ΔY has expectation, variance and covariance

$$\mathsf{E}(\Delta Y) = 0,\ \mathsf{E}(\Delta Y^2) = \frac{1}{3}(\Delta t)^3,\ \mathsf{E}(\Delta Y\Delta W) = \frac{1}{2}(\Delta t)^2 \tag{3.14}$$

($\longrightarrow$ Exercise 3.4). Such a random variable can be realized by two independent normally distributed variates Z_1 and Z_2,

$$\Delta Y = \frac{1}{2}(\Delta t)^{3/2}\left(Z_1 + \frac{1}{\sqrt{3}}Z_2\right) \quad \text{with } Z_i \sim \mathcal{N}(0,1), \quad i = 1,2 \tag{3.15}$$

($\longrightarrow$ Exercise 3.5). With this realization of ΔY we have approximations of $I_{(0,1)}$ and $I_{(1,0)}$, which are substituted into (3.13).

Next the random variable ΔW is replaced by other variates for which the moments match. Choosing $\Delta\widetilde{W}$ trivalued such that the two values $\pm\sqrt{3\Delta t}$ occur with probability 1/6, and the value 0 with probability 2/3, then the random variable $\Delta\widetilde{Y} := \frac{1}{2}\Delta t\ \Delta\widetilde{W}$ has the moments in (3.14) up to terms of order $O(\Delta t^3)$ ($\longrightarrow$ Exercise 3.6). As a consequence, the simplification of (3.13)

$$\begin{aligned} y_{j+1} = y_j + a\Delta t + b\Delta\widetilde{W} + \frac{1}{2}bb'\left((\Delta\widetilde{W})^2 - \Delta t\right) \\ + \frac{1}{2}\left(aa' + \frac{1}{2}b^2a''\right)\Delta t^2 + \frac{1}{2}\left(a'b + ab' + \frac{1}{2}b^2b''\right)\Delta\widetilde{W}\Delta t \end{aligned} \tag{3.16}$$

is second-order weakly convergent.

Higher–Dimensional Cases

In higher-dimensional cases there are additionally mixed terms. We distinguish two kinds of "higher–dimensional":

1.) $y \in \mathbb{R}^n$, a, $b \in \mathbb{R}^n$. Then, for instance, replace bb' by $\frac{\partial b}{\partial y}b$, where $\frac{\partial b}{\partial y}$ is the Jacobian matrix of all first-order partial derivatives.
2.) For multiple Wiener processes the situation is more complicated, because then simple explicit integrals as in (3.9) do not exist. Only the Euler scheme remains simple: for m Wiener processes the Euler scheme is

$$y_{j+1} = y_j + a\Delta t + b^{(1)}\Delta W^{(1)} + \ldots + b^{(m)}\Delta W^{(m)}\,.$$

The Figure 3.2 depicts two components of the system of Example 1.15.

Jump Diffusion

Jump diffusion can be simulated analogously as pure diffusion. Thereby the jump times are not included in the equidistant grid of the $j\Delta t$. An alternative is to simulate the jump times $\tau_1, \tau_2, \ldots$ separately, and superimpose them on the Δt-size grid. Then the jumps can be carried out correctly. With such jump-adapted schemes higher accuracy can be obtained [BrLP06], see also [HiK05].

3.4 Intermediate Values

Integration methods as discussed in the previous section calculate approximations y_j only at the grid points t_j. This leaves the question how to obtain intermediate values, namely, approximations $y(t)$ for $t \neq t_j$. This situation is simple for deterministic ODEs. There we have in general smooth solutions, which suggests to construct an interpolation curve joining the calculated points (y_j, t_j). The deterministic nature guarantees that the interpolation is reasonably close to the exact solution, at least for small steps Δt.

A smooth interpolation is at variance with the stochastic nature of solutions of SDEs. When Δt is small, it may be sufficient to match the "appearance" of a stochastic process. For example, a linear interpolation is easy to be carried out. Such an interpolating continuous polygon was used for the Figures 1.16 and 1.17. Another easily executable alternative would be to construct an interpolating step function with step length Δt. Such an argumentation is concerned with the graphical aspects of filling, and does not pay attention to the law given by an underlying SDE.

The situation is different when the gaps between two calculated y_j and y_{j+1} are large. Then the points that are supposed to fill the gaps should satisfy the underlying SDE. A *Brownian bridge* is a proper means to fill the gaps in Brownian motion. For illustration assume that y_0 (for $t = 0$) and y_T (for $t = T$) are to be connected. Then the Brownian bridge defined by

$$B_t = y_0 \left(1 - \frac{t}{T}\right) + y_T \frac{t}{T} + \left\{ W_t - \frac{t}{T} W_T \right\} \tag{3.17}$$

describes the stochastic behavior that matches Brownian motion. The first two terms represent a straight-line connection between y_0 and y_T. This line segment stands for the trend. The term $W_t - \frac{t}{T} W_T$ describes the stochastic fluctuation (⟶ Exercise 3.7).

Bridges such as the Brownian bridge have important applications. For example, suppose that for a stochastic process S_t a large step has been taken from S_0 to some value S_T. The question may be, what is the largest value of S_t in the gap $0 < t < T$? Or, does S_t reach a certain barrier B? Of course, answers can be expected only with a certain probability. A crude method to tackle the problem would be to calculate a dense chain of S_{t_j} in the gap with a small step size Δt. This is a costly way to get the information. As an alternative, one can evaluate the relevant probabilities of the behavior of bridges directly, without explicitly constructing intermediate points. In this way, larger steps are possible, and costs are reduced. There are several alternative ways to calculate intermediate values, in particular in the multifactor case [Gla04]. For example, the principal component analysis can be applied to approximate the bridge. Here the covariance matrix is taken from the vector $(W(t_0), \ldots, W(t_m))$, where $t_m = T$.

3.5 Monte Carlo Simulation

As pointed out in Section 2.4 in the context of calculating integrals, Monte Carlo is attractive in high-dimensional spaces. The same characterization holds when Monte Carlo (MC) is applied to the valuation of options. For sake of clarity we describe the approach for European vanilla options in context with the one-dimensional Black–Scholes model. But bear in mind that MC is broadly applicable, which will be demonstrated by means of an exotic option at the end of this section.

From Section 1.7.2 we take the one-factor model of a geometric Brownian motion of the asset price S_t,

$$\frac{\mathrm{d}S}{S} = \mu\,\mathrm{d}t + \sigma\,\mathrm{d}W\,.$$

Here μ is the expected growth rate. When options are to be priced we assume a risk-neutral world and replace μ accordingly (by r, or by $r-\delta$ in case of a dividend yield δ, compare Section 1.7.3 and Remark 1.14. Recall the lognormal distribution of GBM, with density function (1.48).

The Monte Carlo simulation of options can be seen in two ways: either dynamically as a process of simulating numerous paths of prices S_t with subsequent appropriate valuation (as suggested by Figure 3.1), or as the formal MC approximation of integrals. For the latter view we briefly recall the integral representation of options in Subsection 3.5.1. Both views are equivalent; the simulation aspect can be seen as financial interpretation and implementation of the MC procedure for integrals.

3.5.1 Integral Representation

In the one-period model of Section 1.5 the valuation of an option was summarized in (1.19) as the discounted values of a probable payoff,

$$V_0 = \mathrm{e}^{-rT}\mathsf{E}_{\mathsf{Q}}(V_T)\,.$$

For the binomial model we prove for European options in Exercise 1.8 that this method produces

$$V_0^{(M)} = \mathrm{e}^{-rT}\mathsf{E}(V_T)\,,$$

where E reflects expectation with respect to the risk-free probability of the binomial method. And for the continuous-time Black–Scholes model, the result in (A4.11b) for a put is

$$V_0 = \mathrm{e}^{-rT}\,\left[K\,F(-d_2) - \mathrm{e}^{(r-\delta)T}S\,F(-d_1)\right]\,, \tag{3.18}$$

similarly for a call. Since F is an integral (⟶ Appendix D2), equation (3.18) is a first version of an integral representation. Its origin is either the analytic solution of the Black–Scholes PDE, or the representation

$$V_0 = \mathrm{e}^{-rT} \int_0^\infty (K - S_T)^+ \, f_{\mathrm{GBM}}(S_T, T;\, S_0, r, \sigma) \, \mathrm{d}S_T \,. \tag{3.19}$$

Here $f_{\mathrm{GBM}}(S_T, T;\, S_0, \mu, \sigma)$ is the density (1.48) of the lognormal distribution, with $\mu = r$, or μ replaced by $r - \delta$ to match a continuous dividend yield δ. It is not difficult to prove that (3.18) and (3.19) are equivalent ($\longrightarrow$ Exercise 3.9 for $\delta = 0$). We summarize the integral representation as

$$\boxed{V(S_0, 0) = \mathrm{e}^{-rT} \mathsf{E}_{\mathsf{Q}}(V(S_T, T) \,|\, S_0)} \tag{3.20}$$

The risk-neutral expectation E_{Q} is explained in Section 1.5. All these expectations are conditional on paths starting at $t = 0$ with the value S_0.

An integral representation offers another way to calculate V_0, namely, via an approximation by means of numerical quadrature methods (see Appendix C1), rather than applying MC. Of course, in this one-dimensional situation, the approximation of the closed-form solution (3.18) is more efficient. But in higher-dimensional spaces integrals corresponding to (3.19) can be become attractive for computational purposes. Note that the integrand is smooth because the zero branch of the put's payoff $(K - S_T)^+$ needs not be integrated; in (3.19) the integration is cut to the interval $0 \le S_T \le K$. Any numerical quadrature method can be applied, such as sparse-grid quadrature [GeG98], [Rei04], [Que07]. But in what follows, we stay with Monte Carlo approximations.

3.5.2 Basic Version for European Options

The simulation aspect of Monte Carlo has been described before, see Figure 3.1. The procedure consists in calculating a large number N of trajectories of the SDE, always starting from S_0, and then average over the payoff values $\Psi((S_T)_k)$ of the samples $(S_T)_k$, $k = 1, \ldots, N$, in order to obtain information on the probable behavior of the process. This is identical to the formal MC method for approximating an integral as (3.19), see Section 2.4. The equivalence with the simulation aspect is characterized by the convergence

$$\frac{1}{N} \sum_{k=1}^{N} \Psi((S_T)_k) \;\longrightarrow\; \int_{-\infty}^{\infty} \Psi(S_T) \, f_{\mathrm{GBM}}(S_T) \, \mathrm{d}S_T = \mathsf{E}(\Psi(S_T)),$$

see (B1.3). The probability distribution of the samples $(S_T)_k$ must match the density of the chosen model, here f_{GBM}. For the Black–Scholes model, these samples are provided by integrating the correct SDE (1.33) under the risk-neutral measure ($\mu = r$ for a non-dividend paying asset). Finally, the result is discounted at the risk-free rate r to obtain the value for $t = 0$.

After having chosen the three items model, current initial value S_0, and payoff function Ψ, the Monte Carlo method works as follows:

Algorithm 3.6 (Monte Carlo simulation of European options)

(1) *For* $k = 1, ..., N$: Choose a seed and integrate the SDE of the underlying model for $0 \le t \le T$ under the risk-neutral measure. (for example, $\mathrm{d}S = rS\,\mathrm{d}t + \sigma S\,\mathrm{d}W$)
Let the final result be $(S_T)_k$.

(2) By evaluating the payoff function Ψ one obtains the values

$$(V(S_T, T))_k := \Psi((S_T)_k), \quad k = 1, ..., N.$$

(3) An estimate of the risk-neutral expectation is

$$\widehat{\mathsf{E}}(V(S_T, T)) := \frac{1}{N} \sum_{k=1}^{N} (V(S_T, T))_k.$$

(4) The discounted variable

$$\widehat{V} := \mathrm{e}^{-rT} \widehat{\mathsf{E}}(V(S_T, T))$$

is a random variable with $\mathsf{E}(\widehat{V}) = V(S_0, 0)$.

In case the underlying receives a continuous dividend yield δ, replace the r in step (1) by $r - \delta$. (not in step (4)!) The resulting $\widehat{V}$ is the desired approximation $\widehat{V} \approx V(S_0, 0)$. In this simple form, the Monte Carlo simulation can only be applied to European options where the exercise date is fixed. Only the value $V(S_0, 0)$ is obtained, and the lack of other information on $V(S, t)$ does not allow to check whether the early-exercise constraint of an American option is violated. For American options a greater effort in simulation is necessary, see Section 3.6. The convergence behavior corresponds to that discussed for Monte Carlo integration, see Section 2.4. In practice the number N must be chosen large, for example, $N = 10000$. This explains why Monte Carlo simulation in general is expensive. For standard European options with univariate underlying that satisfies the Assumption 1.2, the alternative of evaluating the Black–Scholes formula is by far cheaper. But in principle both approaches provide the same result, where we neglect that accuracies and costs are different.

For multivariate options the MC algorithm works analogously, see the example in Section 3.5.5. But the integration of a system of n SDEs clearly has costs depending on n. So the costs of MC depend on n. In practice, this can affect the error. In case the budget in computing time is limited, which is standard for realtime calculations, a limit on the budget will limit the number N of paths, and in turn, the error. If one path costs κ seconds, and the budget for N paths is b seconds, then (2.14a) states that the attainable

error is of the order $\sqrt{\kappa}/\sqrt{b}$. In this sense, $\kappa = O(n)$ does influence the error of MC considerably.

Note that the above Algorithm 3.6 is a crude version of Monte Carlo simulation, which needs to be refined. Since the simulations are independent, the confidence intervals provided by the Central Limit Theorem can be applied (⟶ Appendix B1). In this way, a probabilistic error control is incorporated. Also methods of variance reduction are applied, see Section 3.5.4.

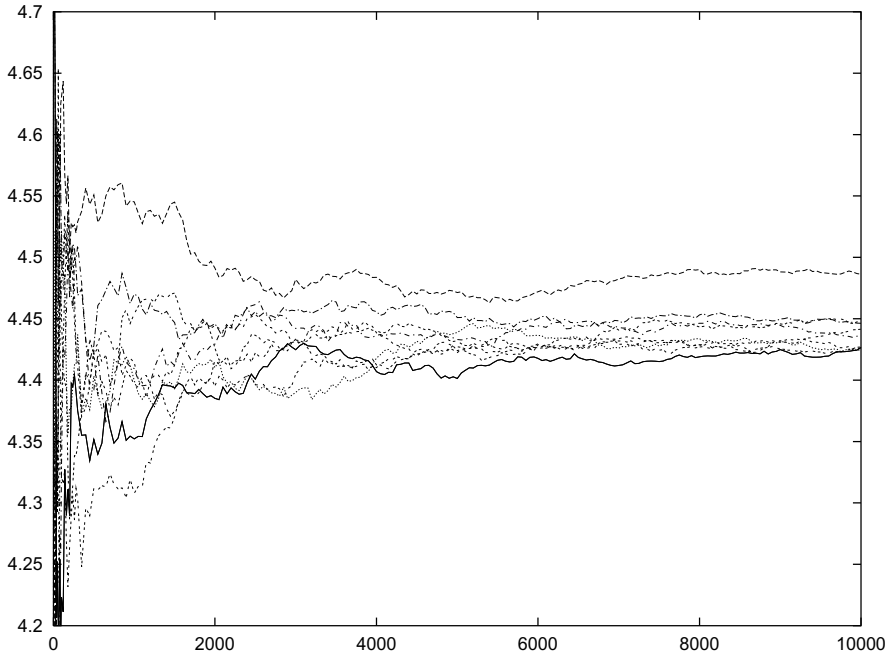

Fig. 3.3. Ten sequences of Monte Carlo simulations on Example 3.7, each with a maximum of 10000 paths. horizontal axis: N, vertical axis: mean value $\widehat{V}$ (suffers from bias, see Section 3.5.3)

Example 3.7 (European put)

Consider a European put with the parameters $S_0 = 5$, $K = 10$, $r = 0.06$, $\sigma = 0.3$, $T = 1$. For the linear SDE $\mathrm{d}S = rS\,\mathrm{d}t + \sigma S\,\mathrm{d}W$ with constant coefficients the theoretical solution is known, see equation (1.54). For the chosen parameters we have

$$S_1 = 5\exp(0.015 + 0.3W_1)\,,$$

which requires "the" value of the Wiener process at $t = 1$. Related values W_1 can be obtained from (1.22) with $\Delta t = T$ as $W_1 = Z\sqrt{T}$, $Z \sim \mathcal{N}(0,1)$. But for this illustration we do not take advantage of the analytic solution formula, because MC is not limited to linear SDEs with constant

coefficients. To demonstrate the general procedure we integrate the SDE numerically with step length $\Delta t < T$, in order to calculate an approximation to S_1. Any of the methods derived in Section 3.3 can be applied. For simplicity we use Euler's method. Since the chosen value of r is small, the discretization error of the drift term is small compared to the standard deviation of W_1. As a consequence, the accuracy of the integration for small values of Δt is hardly better than for larger values of the step size. Artificially we choose $\Delta t = 0.02$ for the time step. Hence each trajectory requires to calculate 50 normal variates $\sim \mathcal{N}(0,1)$. Figure 3.3 shows the values $\widehat{V} \approx V(S_0, 0)$ for 10 sequences of simulations, each with a maximum of $N = 10000$ trajectories, calculated with Algorithm 3.6. Each sequence has started with a different seed for the calculation of the random numbers from Section 2.3.

The Example 3.7 is a European put with the same parameters as Example 1.5. This allows to compare the results of the simulation with the more accurate results from Table 1.2, where we have obtained $V(5, 0) \approx 4.43$. The simulations reported in Figure 3.3 have difficulties to come close to this value. Since Figure 3.3 depicts all intermediate results for sample sizes $N < 10000$, the convergence behavior of Monte Carlo can be observed. For this example and $N < 2000$ the accuracy is bad; for $N \approx 6000$ it reaches acceptable values, and hardly improves for $6000 < N \leq 10000$. Note that the "convergence" is not monotonous, and one of the simulations delivers a frustratingly inaccurate result. ($\longrightarrow$ Exercise 3.11)

3.5.3 Bias

The sampling error of Monte Carlo, which is characterized by the central limit theorem, was already discussed in Section 2.4. Recall the size of this error is proportional to $N^{-1/2}$. In principle, the same error is encountered when Monte Carlo is applied to option valuation. In case of the Black–Scholes model, when the closed-form solution (1.54) of the SDE can be used in step (1) of Algorithm 3.6, the sampling error is basically the only error. But for general options, approximations are often based on discretizations (as in Example 3.7), and some bias is encountered. As a result, the error deteriorates.

Bias typically occurs when the option is path-dependent —that is, its value depends on S_t for possibly all $t \leq T$. For example, the volatility may be local, which means that it depends on S_t, $\sigma = \sigma(S)$. Another example is furnished by the *lookback* option, where the valuation depends on

$$x := \mathsf{E}\left(\max_{0 \leq t \leq T} S_t\right) .$$

In both examples, a time discretization may help with a finite number m of values S_{t_j}, with the notation as used in (3.1). Even if the underlying SDE is

such that a closed-form solution is available, the estimator provided by the discretely sampled maximum

$$\hat{x} := \max_{0 \le j \le m} S_{t_j}$$

almost surely underestimates x. That is, the estimator $\hat{x}$ of x is biased, with

$$\text{bias}(\hat{x}) := \mathsf{E}(\hat{x}) - x \neq 0\,. \tag{3.21}$$

The lookback option is one example where local information on the individual paths is required. Other examples of exotic options requiring S_{t_j} for several t_j are barrier options, and Asian options, see Section 6.1. In these examples, if applied to the Black–Scholes model, the analytic solution can be used locally in each step. Two alternatives for a step from t to $t + \Delta t$ are

$$\begin{aligned} S_{t+\Delta t} &= S_t \exp[(\mu - \tfrac{1}{2}\sigma^2)\,\Delta t + \sigma\,\Delta W] \qquad \text{(unbiased)} \\ S_{t+\Delta t} &= S_t\,(1 + \mu\,\Delta t + \sigma\,\Delta W) \qquad \text{(Euler's step, biased)} \end{aligned} \tag{3.22}$$

For the bias due to the application of Euler's scheme, see Exercise 3.10. Compare Figures 3.3 and 3.5 for results with and without bias.

Fortunately, when sufficient computing time is available, this bias can be made arbitrarily small by taking sufficiently large values of m. There is a tradeoff between making the variance small ($N \to \infty$), and making the bias small ($m \to \infty$, $\Delta t \to 0$). The mean square error

$$\text{MSE}(\hat{x}) := \mathsf{E}[(\hat{x} - x)^2] \tag{3.23a}$$

measures both errors: A straightforward calculation (which the reader may check) shows

$$\begin{aligned} \text{MSE}(\hat{x}) &= \left(\mathsf{E}(\hat{x}) - x\right)^2 + \mathsf{E}[(\hat{x} - \mathsf{E}(\hat{x}))^2] \\ &= (\text{bias}(\hat{x}))^2 + \mathsf{Var}(\hat{x}) \end{aligned} \tag{3.23b}$$

The final aim is to make MSE small, and the investigator must balance the effort in controlling the bias or the sampling error.

We outline this for a Monte Carlo approximation that makes use of a numerical integration scheme such as Euler's method. For brevity, write again h for the step Δt. Let $\hat{x} := y_T^h$ be the result of a weakly convergent discretization scheme, see Definition 3.4, with order β and g =identity. Then the bias of the discretization is of the order β,

$$\text{bias}(\hat{x}) = \alpha_1 h^\beta\,, \quad \alpha_1 \text{ a constant.}$$

Since the variance of Monte Carlo is of the order N^{-1} (N the sample size, see (2.14a)), (3.23b) leads to model the mean square error as

$$\text{MSE} = \alpha_1^2 h^{2\beta} + \frac{\alpha_2}{N}$$

for some constant α_2. This error model allows to analyze the tradeoff ($N \to \infty$ or $h \to 0$) more closely ($\longrightarrow$ Exercise 3.13). It turns out that for optimally chosen h, N the error $\sqrt{\text{MSE}}$ behaves like

$$\sqrt{\text{MSE}} \sim C^{-\frac{\beta}{1+2\beta}}$$

where C denotes the costs of the approximation. Applying Euler's method ($\beta = 1$) gives the exponent $-1/3$, clearly worse than the exponent $-1/2$ of an unbiased Monte Carlo. As [Gla04] points out, this result emphasizes the importance of high-order schemes ($\beta > 1$) for high demands of accuracy.

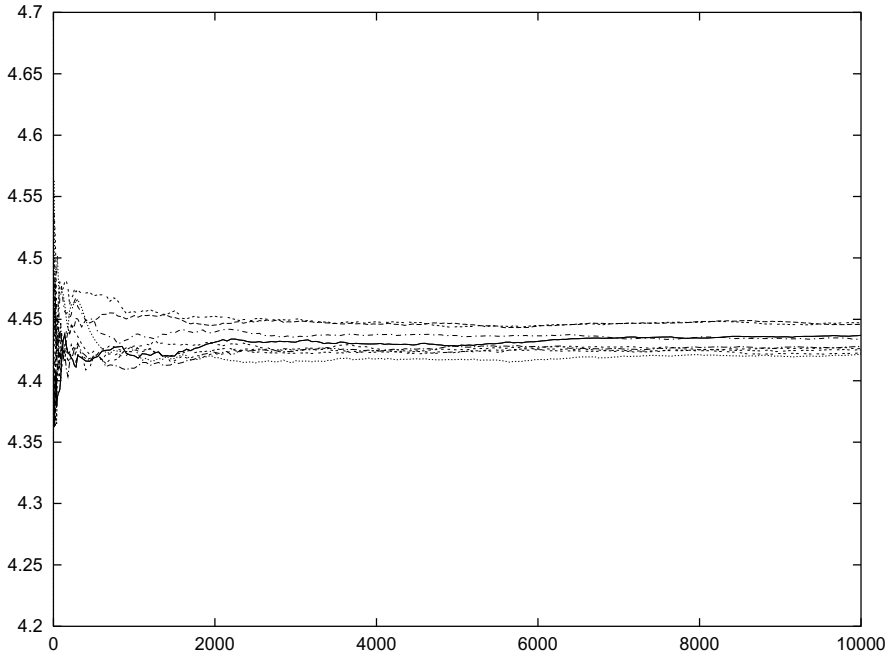

Fig. 3.4. Ten series of antithetic simulations on Example 3.7

3.5.4 Variance Reduction

To improve the accuracy of simulation and thus the efficiency, it is essential to apply methods of variance reduction. We explain the methods of the *antithetic variates* and the *control variates*. In many cases these methods decrease the variances.

Antithetic Variates

If a random variable satisfies $Z \sim \mathcal{N}(0,1)$, then also $-Z \sim \mathcal{N}(0,1)$. Let $\widehat{V}$ denote the approximation obtained by Monte Carlo simulation. With little extra effort during the original Monte Carlo simulation we can run in parallel

a side calculation which uses $-Z$ instead of Z. For each original path this creates a "partner" path, which looks like a mirror image of the original. The partner paths also define a Monte Carlo simulation of the option, called the *antithetic variate*, denoted by V^-. The average

$$V_{\mathrm{AV}} := \tfrac{1}{2}\left(\widehat{V} + V^-\right) \tag{3.24}$$

(AV for *antithetic variate*) is a new approximation, which in many cases is more accurate than $\widehat{V}$. Since $\widehat{V}$ and V_{AV} are random variables we can only aim at

$$\mathsf{Var}(V_{\mathrm{AV}}) < \mathsf{Var}(\widehat{V}) \ .$$

In view of the properties of variance and covariance (equation (B1.7)),

$$\begin{aligned} \mathsf{Var}(V_{\mathrm{AV}}) &= \tfrac{1}{4}\mathsf{Var}(\widehat{V} + V^-) \\ &= \tfrac{1}{4}\mathsf{Var}(\widehat{V}) + \tfrac{1}{4}\mathsf{Var}(V^-) + \tfrac{1}{2}\mathsf{Cov}(\widehat{V}, V^-)\,. \end{aligned} \tag{3.25}$$

From

$$|\mathsf{Cov}(X,Y)| \le \frac{1}{2}[\mathsf{Var}(X) + \mathsf{Var}(Y)]$$

(follows from (B1.7)) we deduce

$$\mathsf{Var}(V_{\mathrm{AV}}) \le \frac{1}{2}(\mathsf{Var}(\widehat{V}) + \mathsf{Var}(V^-))\,.$$

By construction, $\mathsf{Var}(\widehat{V}) = \mathsf{Var}(V^-)$ should hold. Hence $\mathsf{Var}(V_{\mathrm{AV}}) \le \mathsf{Var}(\widehat{V})$. This shows that in the worst case only the efficiency is slightly deteriorated by the additional calculation of V^-. The favorable situation is when the covariance is negative. Then (3.25) shows that the variance of V_{AV} can become significantly smaller than that of $\widehat{V}$. Since we have chosen the random numbers $-Z$ for the calculation of V^-, the chances are high that $\widehat{V}$ and V^- are negatively correlated and hence $\mathsf{Cov}(\widehat{V}, V^-) < 0$. In this situation V_{AV} is a better approximation than $\widehat{V}$. Variance reduction by antithetic variates may not be too effective, but is easily implemented.

In Figure 3.4 we simulate Example 3.7 again, now with antithetic variates. With this example and the chosen random number generator[2] the variance reaches small values already for small N. Compared to Figure 3.3 the convergence is somewhat smoother. The accuracy the experiment shown in Figure 3.3 reaches with $N = 6000$ is achieved already with $N = 2000$ in Figure 3.4. But in the end, the error has not become really small. The main reason for the remaining significant error in the experiment reported by Figure 3.4 is the bias due to the discretization error of the Euler scheme. To remove this source of error, we repeat the above experiments with the analytical solution of (1.54). The result is shown in Figure 3.5 for crude Monte Carlo, and in

[2] the simple generator of Algorithm 2.7

Figure 3.6 for MC with antithetic variates. These figures better reflect the convergence behavior of Monte Carlo simulation. By the way, applying the Milstein scheme of Algorithm 3.5 does not improve the picture: No qualitative change is visible if we replace the Euler-generated simulations of Figures 3.3/3.4 by their Milstein counterparts. This may be explained by the fact that the weak convergence order of Milstein's method equals that of the Euler method. — Recall that Example 3.7 is chosen merely for illustration; here other methods are by far more efficient than Monte Carlo approaches.

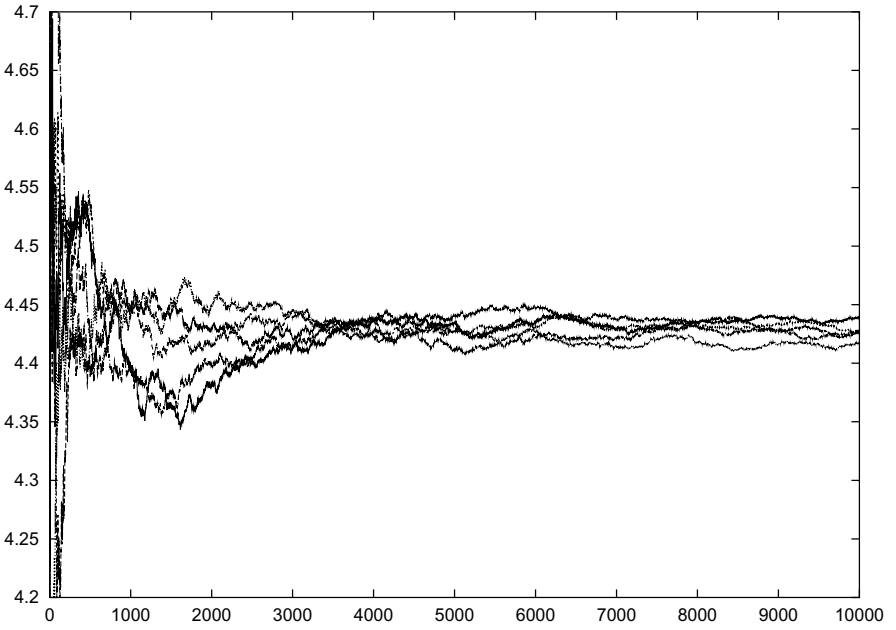

Fig. 3.5. Five series of Monte Carlo simulations on Example 3.7, using the analytic solution of the SDE (compare to Fig. 3.3)

Control Variates

Again V denotes the exact value of the option and $\widehat{V}$ a Monte Carlo approximation. For comparison we calculate in parallel another option, which is closely related to the original option, and for which we know the exact value V^*. Let the Monte Carlo approximation of V^* be denoted $\widehat{V}^*$. This variate serves as *control variate* with which we wish to "control" the error. The additional effort to calculate the control variate $\widehat{V}^*$ is small in case the simulations of the asset S are identical for both options. This situation arises when S_0, μ and σ are identical and only the payoff differs. When the two options are similar enough one may expect a strong positive correlation between them. So we expect relatively large values of $\mathsf{Cov}(V, V^*)$ or $\mathsf{Cov}(\widehat{V}, \widehat{V}^*)$, close to its upper bound,

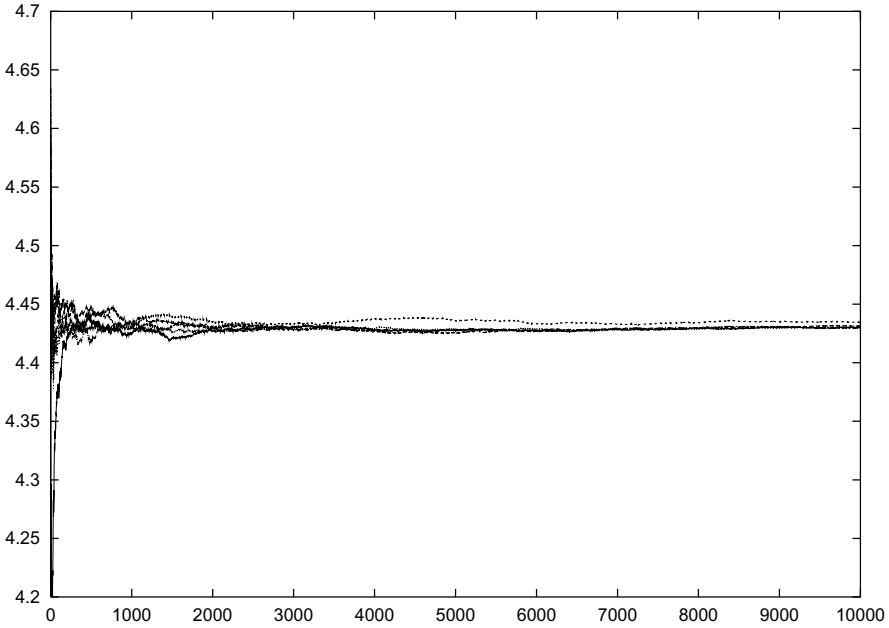

Fig. 3.6. Five series of Monte Carlo simulations on Example 3.7 using the analytic solution of the SDE and antithetic variates (3.24) (compare to Fig. 3.4)

$$\mathsf{Cov}(\widehat{V},\widehat{V}^*) \approx \frac{1}{2}\mathsf{Var}(\widehat{V}) + \frac{1}{2}\mathsf{Var}(\widehat{V}^*)\,.$$

This leads us to define "closeness" between the options as sufficiently large covariance in the sense

$$\mathsf{Cov}(\widehat{V},\widehat{V}^*) > \frac{1}{2}\mathsf{Var}(\widehat{V}^*)\,. \tag{3.26}$$

The method is motivated by the assumption that the unknown error $V-\widehat{V}$ has the same order of magnitude as the known error $V^*-\widehat{V}^*$. This anticipation can be written $V \approx \widehat{V} + (V^*-\widehat{V}^*)$, and leads to define

$$V_{\mathrm{CV}} := \widehat{V} + V^* - \widehat{V}^* \tag{3.27}$$

as another approximation (CV for *control variate*). We see from (B1.6) (with $\beta = V^*$) and (B1.7) that

$$\mathsf{Var}(V_{\mathrm{CV}}) = \mathsf{Var}(\widehat{V}-\widehat{V}^*) = \mathsf{Var}(\widehat{V}) + \mathsf{Var}(\widehat{V}^*) - 2\mathsf{Cov}(\widehat{V},\widehat{V}^*)\,.$$

If (3.26) holds, then $\mathsf{Var}(V_{\mathrm{CV}}) < \mathsf{Var}(\widehat{V})$. In this sense $\mathsf{Var}(V_{\mathrm{CV}})$ is a better approximation than $\widehat{V}$.

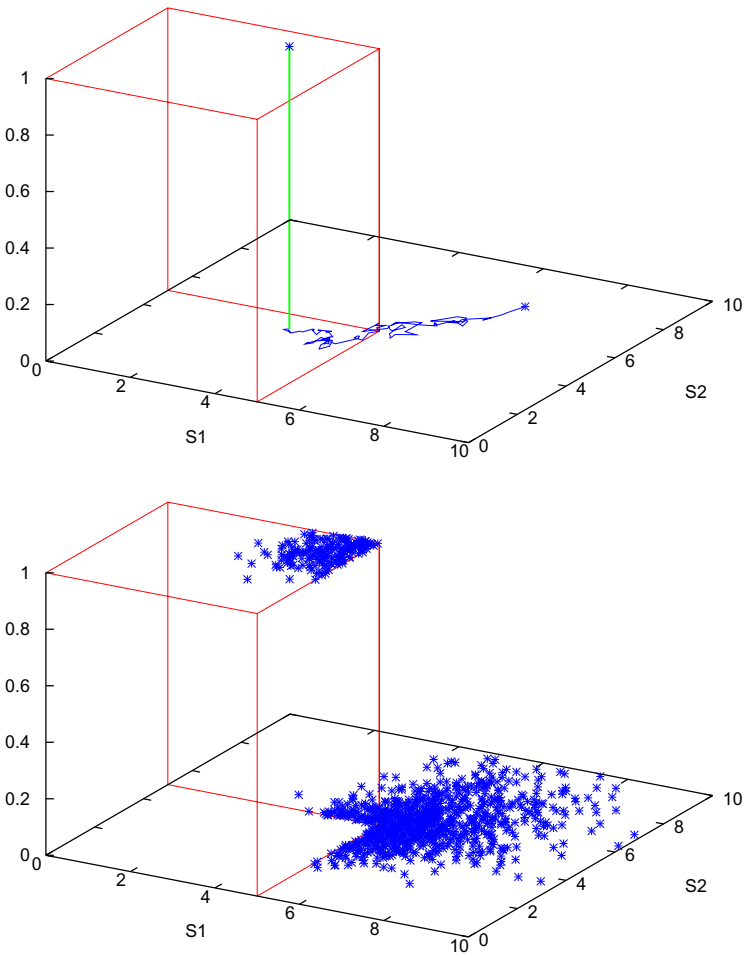

Fig. 3.7. Example 3.8, binary option. horizontal: (S_1, S_2)-plane, vertical: $V(S_1, S_2)$; top: two paths starting at $S_1 = S_2 = 5$ with their payoff values; bottom: $N = 1000$ terminal points with their payoff values

3.5.5 Application to an Exotic Option

As mentioned before, the error of Monte Carlo methods basically does not vary with the dimension. As an example we choose a two-dimensional binary put to illustrate that MC can be applied as easily as in a one-dimensional situation.

Assume that two underlying assets $S_1(t), S_2(t)$ obey a two-dimensional GBM,

$$\begin{aligned} \mathrm{d}S_1 &= S_1\,(\mu_1\,\mathrm{d}t + \sigma_1\,\mathrm{d}W^{(1)}) \\ \mathrm{d}S_2 &= S_2\,(\mu_2\,\mathrm{d}t + \sigma_2\,(\rho\,\mathrm{d}W^{(1)} + \sqrt{1-\rho^2}\,\mathrm{d}W^{(2)}))\,. \end{aligned} \tag{3.28}$$

This makes use of Exercise 2.9: $W^{(1)}$ and $W^{(2)}$ are two uncorrelated Wiener processes, and the way they interact in (3.28) establishes a correlation ρ between S_1 and S_2. The analytic solution of (3.28) is given by

$$\begin{aligned} S_1(T) &= S_1(0)\exp\left((\mu_1 - \frac{1}{2}\sigma_1^2)T + \sigma_1 W^{(1)}(T)\right) \\ S_2(T) &= S_2(0)\exp\left((\mu_2 - \frac{1}{2}\sigma_2^2)T + \sigma_2(\rho W^{(1)}(T) + \sqrt{1-\rho^2}\,W^{(2)}(T))\right)\,, \end{aligned} \tag{3.29}$$

which generalizes (1.54).

Example 3.8 (2D European binary put)
A two-asset cash-or-nothing put pays the fixed cash amount c in case

$$S_1(T) < K_1 \quad \text{and} \quad S_2(T) < K_2\,.$$

We choose the parameters $T = 1$, $K_1 = K_2 = 5$, $\sigma_1 = 0.2$, $\sigma_2 = 0.3$, $\rho = 0.3$, $c = 1$, $r = 0.1$; no dividends, so the "costs of carry" are taken as $\mu_1 = \mu_2 = r$. The value $V(S_1, S_2, 0)$ is to be evaluated at $S_1(0) = S_2(0) = 5$.

Figure 3.7 illustrates both the payoff of this exotic option and the Monte Carlo approach. The top figure depicts the box characterizing the payoff. Further, two paths starting at $S_1(0) = S_2(0) = 5$ are drawn. For $t = T$, one of the paths ends inside the box; accordingly the payoff value there is $V = c = 1$. The other path terminates "outside the strike," the payoff value is zero. Since we have the analytic solution (3.29), no paths need to be calculated. Rather, terminal points $(S_1(T), S_2(T))$ are evaluated by (3.29). The lower figure in Figure 3.7 shows 1000 points calculated in this way. Taking the mean value and discounting as in Algorithm 3.6, yields approximations to $V(5, 5, 0)$. With $N = 10^5$ simulations we obtain

$$V(5, 5, 0) \approx 0.174\,,$$

using random numbers based on the simple generator of Algorithm 2.7. The accuracy is almost three digits.[3] Using Euler's method rather than the analytic solution, Example 3.8 offers nice possibilities to conduct empirical studies in controlling either the bias or the sample error. We conclude Example 3.8 with Figure 3.8, which depicts the entire surface $V(S_1, S_2, 0)$, calculated with Algorithm 1.18 [Que07].

[3] This example has an analytic solution based on bivariate distribution functions, see [Haug98].

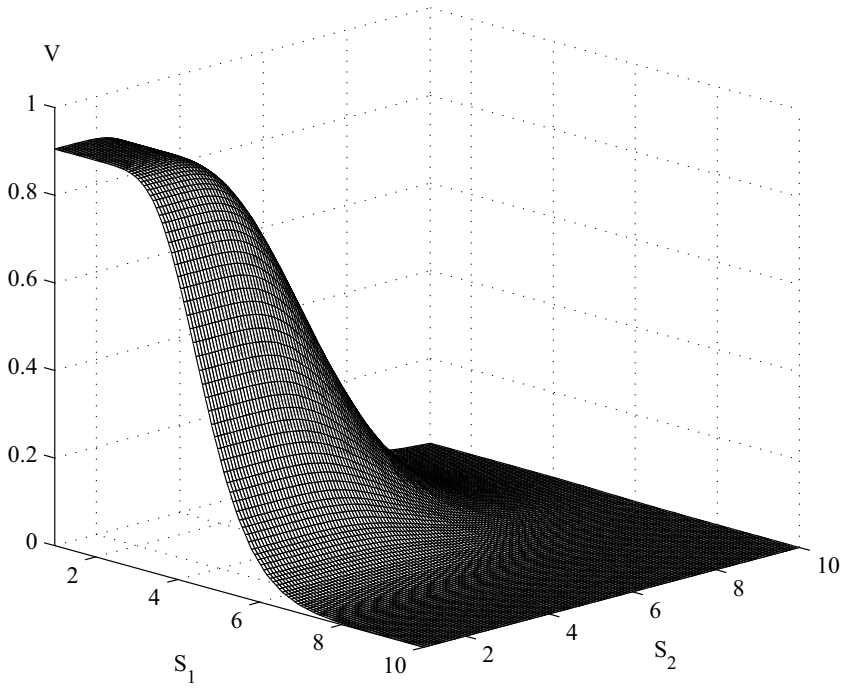

Fig. 3.8. Example 3.8: surface $V(S_1, S_2, 0)$ calculated by Algorithm 1.18. With kind permission of Sebastian Quecke.

3.6 Monte Carlo Methods for American Options

The equation (3.20) can be generalized to American options. Similar as for European options, Monte Carlo applied to American options requires simulating paths S_t of the underlying model. Again, for ease of exposition, we think of the prototype example of the univariate Black–Scholes model where we integrate $\mathrm{d}S_t = rS_t\,\mathrm{d}t + \sigma S_t\,\mathrm{d}W_t$ for $t \geq 0$. Whereas for European options it is clear to integrate until expiration, $t = T$, the American option requires to continuously investigate whether early exercise is advisable.

3.6.1 Stopping Time

For motivation, think of a price limit β of an asset, such that a stop-buy order is to be carried out at that level. The decision is prompted by the event that S_t reaches β for some "stopping time" τ. Or, for the life of an American option, the decisive event is "early exercise," which amounts to a "stop" in holding the option. To mimic reality, one must take care that for any t the

decision (on early exercise, for example) is *only based on the information that is known so far.* This situation suggests defining a *stopping time* to be not anticipating. A stochastic process S_t builds a natural filtration $\mathcal{F}_t$, which is interpreted as a model of the information available at time t (⟶ Appendix B2). Accordingly, for a stopping time τ we require $\{\tau \le t\} \in \mathcal{F}_t$ for all $t \ge 0$, where the set $\{\tau \le t\}$ represents all decisions until time t. That is:

Definition 3.9 (stopping time)
A stopping time τ with respect to a filtration $\mathcal{F}_t$ is a random variable that is $\mathcal{F}_t$-measurable for all $t \ge 0$.

Typically, a decision is triggered when τ is reached, such as exercising early. For any time t we know whether $\tau \le t$ —that is, whether the decision is made. Suppose we travel along the path of a specific realization of a stochastic process S_t and look up at the event that defines τ. In this way we get a realization of the random variable τ; for each path obtain another value.

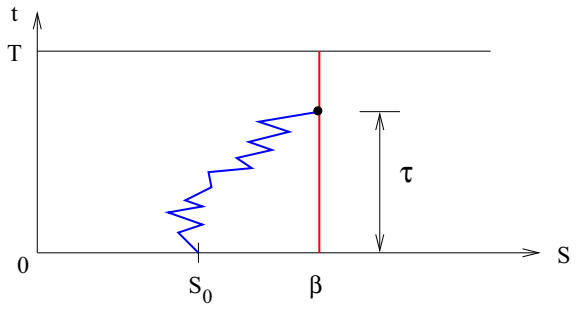

Fig. 3.9. The strategy of Example 3.10 to define a stopping time τ

Two examples should make the concept of a stopping time clearer.

Example 3.10 (hitting time)
For a value β, which fixes a level of S, define

$$\tau := \inf\{\, t > 0 \;\mid\; S_t \ge \beta \,\}\,,$$
$$\text{and } \tau := \infty \quad \text{if such a } t \text{ does not exist.}$$

This example, illustrated in Figure 3.9, fulfills the requirements of a stopping time.[4] It defines a *stopping strategy*, "stop when S_t has reached β."

The example

$$t^* := \text{ moment when } S_t \text{ reaches its maximum over } 0 \le t \le T$$

is no stopping time, because for each $t < T$ it can not be decided whether $t^* \le t$ or $t^* > t$; it is not possible to decide whether to stop.

[4] For a proof see [HuK00], p.42, or [Shr04], p.341.

In the context of American options, of all possible stopping times, the stopping at the early-exercise curve is optimal (illustrated in Figure 3.10). This optimal stopping gives the American option its optimal value. From a practical point of view, the stopping at the early-exercise curve can not be established as in Example 3.10, because the curve is not known initially. But the following characterization of the value $V(S,0)$ of an American option holds true:

$$V(S,0) = \sup_{0\le\tau\le T} \mathsf{E}_{\mathsf{Q}}(\mathrm{e}^{-r\tau}\,\Psi(S_\tau) \mid S_0 = S)\,, \qquad (3.30)$$

where τ is a stopping time and Ψ is the payoff.

This result is a special case for $t = 0$ of a more general formula for $V(S,t)$, which is proved in [Ben84]. Clearly, (3.30) includes the case of a European option for $\tau := T$, in which case taking the supremum is not effective.

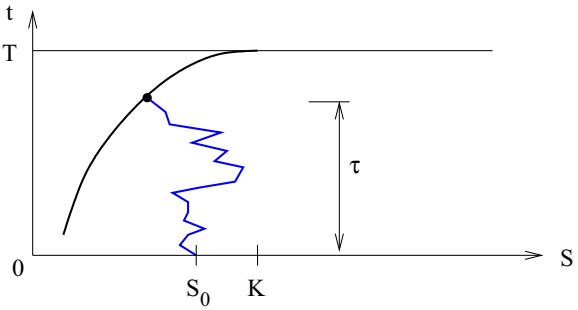

Fig. 3.10. The optimal stopping time τ of a vanilla put. The heavy curve is the early-exercise curve, and the zigzag symbolizes a path S_t.

3.6.2 Parametric Methods

A practical realization of (3.30) leads to calculating lower bounds $V^{\mathrm{low}}(S,0)$ and upper bounds $V^{\mathrm{up}}(S,0)$ such that

$$V^{\mathrm{low}}(S,0) \le V(S,0) \le V^{\mathrm{up}}(S,0)\,. \qquad (3.31)$$

Since by (3.30) $V(S,0)$ is given by taking the supremum over *all* stopping times, a lower bound is obtained by taking a *specific* stopping strategy. To illustrate the idea, choose the stopping strategy of Example 3.10 with a level β, see Figure 3.9. If we denote for each calculated path the resulting stopping time by $\tilde\tau = \tilde\tau(\beta)$, a lower bound to $V(S,0)$ is given by

$$V^{\mathrm{low}(\beta)}(S,0) := \mathsf{E}_{\mathsf{Q}}(\mathrm{e}^{-r\tilde\tau}\,\Psi(S_{\tilde\tau}) \mid S_0 = S)\,. \qquad (3.32)$$

This value depends on the parameter β, which is indicated by writing $V^{\mathrm{low}(\beta)}$. The bound is calculated by Monte Carlo simulation over a sample of N paths, where the paths are stopped according to the chosen stopping rule. Procedure

and costs of such a simulation for one value of β are analogous as in Algorithm 3.6. Repeating the experiment for another value of β may produce a better (larger) value $V^{\mathrm{low}(\beta)}$.

It is difficult to get a tolerable accuracy when working with only a single parameter β. The situation can be slightly improved by choosing a finishing line different from Figure 3.9. A simple but nicely working approximation uses a parabola in the (S,t)-domain with horizontal tangent at $t=T$. Again this approach requires only one parameter β ($\longrightarrow$ Exercise 3.12). A result of this approach is illustrated in Figure 3.11.

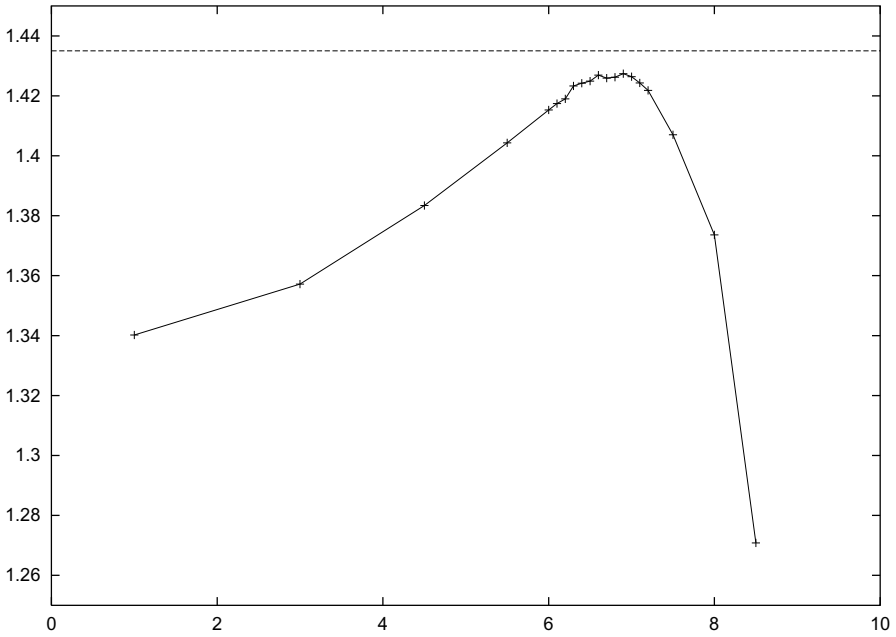

Fig. 3.11. Monte Carlo approximations $V^{\mathrm{low}(\beta)}(S,0)$ (+) for several values of β (Exercise 3.12, random numbers from [MaN98]). The dashed line represents the exact value $V(S,0)$.

There are many examples how to obtain better lower bounds. For instance, the early-exercise curve can be approximated by pieces of curves or pieces of straight lines, which are defined by several parameters; β then symbolizes a vector of parameters. The idea is to optimize in the chosen parameter space, trusting that

$$\sup_{\beta} V^{\mathrm{low}(\beta)} \approx V.$$

As illustrated by Figure 3.11, the corresponding surface to be maximized is not smooth. Accordingly, an optimization in the parameter space is costly, see Appendix C4. Recall that each evaluation of $V^{\mathrm{low}(\beta)}$ for one β is expensive.

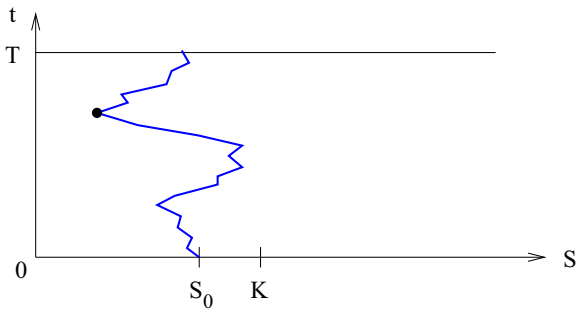

Fig. 3.12. No stopping time; maximizing the payoff of a *given* path

What kind of parametric approximation, and what choice of the parameters can be considered "good" when $V(S,t)$ is still unknown? To this end, upper bounds V^{up} can be constructed, and one attempts to push the difference $V^{\text{up}} - V^{\text{low}}$ close to zero in order to improve the approximation provided by (3.31).[5] An upper bound can be obtained, for example, when one peers into the future. As a crude example, the entire path S_t for $0 \le t \le T$ may be simulated, and the option is "exercised" in retrospect when

$$\mathrm{e}^{-rt}\,\Psi(S_t)$$

is maximal. This is illustrated in Figure 3.12. Pushing the lower bounds $V^{\text{low}(\beta)}$ towards upper bounds amounts to search in the β-parameter space for a better combination of β-values. As a by-product of approximating $V(S,0)$, the corresponding parameters β provide an approximation of the early-exercise curve.

The above is just a crude strategy how Monte Carlo can be applied to approximate American options. In particular, the described simple approach to obtain upper bounds is not satisfactory. Consult [AnB04] for a systematic way of constructing reasonable upper bounds. Typically, the upper bounds are more costly than the lower ones. Bounds are also provided by the stochastic grids of [BrG04].

3.6.3 Regression Methods

One basic idea of regression methods is to approximate the American-style option by a Bermudan option. A Bermudan option restricts early exercise to specified discrete dates during its life. As in Section 1.8.4, the time instances with the right to exercise are created artificially by a finite set of discrete time instances t_i :

[5] Since the bounds are approximated by stochastic methods, it may happen that the true value $V(S,0)$ is not inside the calculated interval (3.31).

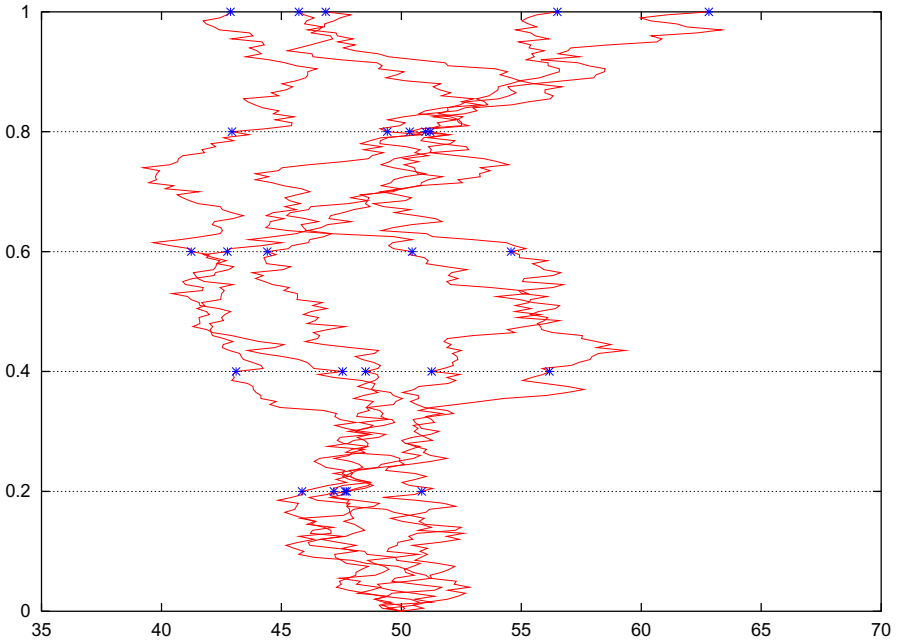

Fig. 3.13. Setting for a Bermudan option; schematic illustration with five trajectories and $M = 5$ exercise times; data as in Figure 3.1; horizontal axis: S, vertical axis: t. The points $(S_{i,k}, t_i)$ are marked.

$$\Delta t := \frac{T}{M}\,, \quad t_i := i\,\Delta t \quad (i = 0, \ldots, M)\,,$$

see the illustration of Figure 3.13. The situation resembles the time discretization of the binomial method of Section 1.4. In that semidiscretized setting the value of the dynamic programming procedure of equation (1.14) generalizes to

$$V_i(S) = \max\{\,\Psi(S)\,,\; V_i^{\mathrm{cont}}(S)\}\,,$$

where the continuation value or *holding value* V_i^{cont} is defined by the conditional expectation

$$V_i^{\mathrm{cont}}(S) := \mathrm{e}^{-r\Delta t}\,\mathsf{E}_{\mathsf{Q}}(V_{i+1}(S_{i+1}) \mid S_i = S)\,.$$

[On the binomial tree, this is equation (1.13).] E_{Q} is calculated as before under the assumption of risk neutrality.

In the context of a Bermudan option, we define the continuation value

$$C_i(x) := \mathrm{e}^{-r\Delta t}\,\mathsf{E}_{\mathsf{Q}}(V(S_{t_{i+1}}, t_{i+1}) \mid S_{t_i} = x)\,. \tag{3.33}$$

This function needs to be approximated. If we can do it, then the general recursion is:

Principle 3.11 (dynamic programming)
Set $V_M(x) = \Psi(x)$. For $i = M-1, ..., 1$
calculate $C_i(x)$ for $x > 0$ and
$V_i(x) := V(x, t_i) = \max\{\Psi(x),\, C_i(x)\}$
$V_0 := V(S_0, 0) = \max\{\Psi(S_0),\, C_0(S_0)\}$

To calculate an approximation $\hat{C}_i(x)$ for $C_i(x)$, data are generated by running N simulations. All simulating paths are calculated starting from S_0, according to the underlying risk-neutral model. This creates paths $S_1(t), \ldots, S_N(t)$ for $0 \le t \le T$ ($N = 5$ in Figure 3.13). At the discrete t_i values, this establishes $S_{i,k} := S_k(t_i)$ and assigns $(S_{i,k}, t_i)$ to $(S_{i+1,k}, t_{i+1})$ for $k = 1, \ldots, N$ and all i. Dropping the index k, this amounts to the transition $S_i \longrightarrow S_{i+1}$. On S_{i+1} a valuation V_{i+1} is calculated by the recursion. Hence N pairs $(S_i, \mathrm{e}^{-r\Delta t} V_{i+1})$ are provided for each i. These pairs match (3.33) and form the data basis on which $(x, C(x))$ is approximated by a suitable minimization method such as least squares.[6] This sets up the basic principle of regression methods.

Algorithm 3.12 (regression I)
(a) Simulate N paths $S_1(t), ..., S_N(t)$. Calculate and store the values

$$S_{i,k} := S_k(t_i)\ , \quad i = 1, ..., M,\ k = 1, ..., N\,.$$

(b) For $i = M$ set $V_{M,k} := \Psi(S_{M,k})$ for all k.
(c) For $i = M-1, ..., 1$:
Approximate $C_i(x)$ using suitable basis functions $\phi_0, ..., \phi_L$ (monomials, for example)

$$C_i(x) \approx \sum_{l=0}^{L} a_l \phi_l(x) =: \hat{C}_i(x)$$

by least squares over the N points

$$(x_k,\, y_k) := (S_{i,k},\, \mathrm{e}^{-r\Delta t} V_{i+1,k})\ , \quad k = 1, ..., N,$$

and set

$$V_{i,k} := \max\left\{\Psi(S_{i,k}),\, \hat{C}_i(S_{i,k})\right\}\ .$$

(d) $\hat{C}_0 := \mathrm{e}^{-r\Delta t} \frac{1}{N}(V_{1,1} + ... + V_{1,N})\,, \quad V_0 = \max\left\{\Psi(S_0),\, \hat{C}_0\right\}$

In step (c), the coefficients $a_0, \ldots, a_L$ of the approximation $\hat{C}$ result from a minimization. Step (d) is needed because (c) does not work for $i = 0$ since all $S_{0,k} = S_0$. In case the S and the x are vectors, the algorithm also describes the multifactor case. Note that for convergence both N and L must be increased.

[6] For least squares see Appendix C4.

The above basic version of regression can be improved in several ways. [LonS01] has introduced a special version of the regression, incorporating as a subalgorithm the calculation of the stopping time of each path. Working with individual stopping times enables to set up an interleaving mechanism over the time levels for comparing cash flows. The central step in (c) changes to

$$V_{i,k} := \begin{cases} \Psi(S_{i,k}) & \text{for } \Psi(S_{i,k}) \geq \hat{C}_i(S_{i,k}) \\ V_{i+1,k} & \text{for } \Psi(S_{i,k}) < \hat{C}_i(S_{i,k}) \end{cases} \tag{3.34}$$

This requires to adapt steps (b), (c), (d). Points out-of-the-money do not enter the regression. To save storage, intermediate values can be filled in by using a bridging technique. Following [Jon09], a significant speed-up is possible when working with a cash-flow vector g, and an integer stopping time vector τ (the integer factors k of $\tau_k = k\Delta t$). The resulting algorithm is:

Algorithm 3.13 (regression II)

(a) Simulate N paths as in Algorithm 3.12.

(b) Set $g_k := \Psi(S_{M,k})$, $\tau_k = M$ for $k = 1, ..., N$.

(c) For $i = M-1, ..., 1$:

For the subset of in-the-money-points

$$(x_k,\, y_k) := (S_{i,k},\, \mathrm{e}^{-r(\tau_k - i)\Delta t} g_k)\,,$$

approximate $C_i(x)$ by $\hat{C}_i(x)$,
and for those k with $\Psi(S_{i,k}) \geq \hat{C}_i(S_{i,k})$: update

$$g_k := \Psi(S_{i,k}),\ \tau_k := i\,.$$

(d) $\hat{C}_0 := \dfrac{1}{N} \displaystyle\sum_{k=1}^{N} \mathrm{e}^{-r\tau_k \Delta t} g_k\ , \quad V_0 := \max\{\Psi(S_0),\, \hat{C}_0\}.$

Figure 3.14 shows a simple setting as an attempt to illustrate the regression method, with strike $K = 10$, and $M = 2$, $N = 5$. For $i = 1$, four of the paths are in the money. Their continuation values $V_{i+1,k}$ are denoted a, b, c, d in Figure 3.14. The heavy line is the regression $\hat{C}$, here a straight line because it is based only on the two regressors $\phi_0 = 1$, $\phi_1 = x$. The maximum $\max\{\Psi, \hat{C}\}$ is easy to check: for the points a and b the payoff is larger than $\hat{C}(S)$.

Recently, many refined Monte Carlo methods for the calculation of American options have been suggested. For an overview on related approaches, consult Chapter 8 in [Gla04]. At current state, the robust regression of [Jon11] appears to be the most efficient approach; it has priced options on baskets of up to 30 assets. One basic ingredient of this method is to neglect outliers, with the effect of a remarkable bias reduction.

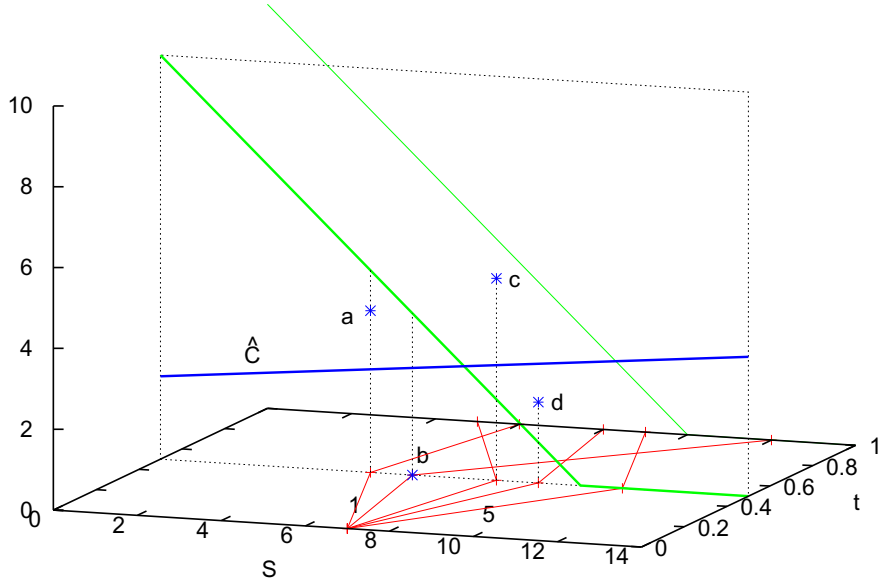

Fig. 3.14. Regression; illustration for a put with $r = 0$, $M = 2$, $K = 10$

3.7 Accuracy, and Sensitivity

Monte Carlo simulation is of great importance for general models where no specific assumptions (as those of Black, Merton and Scholes) have led to efficient approaches. For example, in case the interest rate r cannot be regarded as constant but is modeled by some SDE (such as equation (1.40)), then a system of SDEs must be integrated. Examples of stochastic volatility are provided by Example 1.15, compare Figure 3.2, or by the Heston model (1.43). In such cases, a Monte Carlo simulation can be the method of choice. Then the Algorithm 3.6 is adapted appropriately. Monte Carlo methods are especially attractive for multifactor models with high dimension.

The demands for **accuracy** of Monte Carlo simulation should be kept on a low level. In many cases an error of 1% must suffice. Recall that it does not make sense to decrease the Monte Carlo sampling error significantly below the error of the time discretization of the underlying SDE (and vice versa). When the amount of available random numbers is too small or its quality poor, then no improvement of the error can be expected. The methods of variance reduction can save a significant amount of costs [BoBG97], [ScH97], [Pla99]. Note that different variance-reduction techniques can be combined with each other. The efficiency of Monte Carlo simulations can be enhanced by suitably combining several discretizations with different levels of coarseness [Gil08].

Sensitivity

A great computational challenge is to estimate how the price V changes when parameters or initial states change, see Section 1.4.6. A sensitivity analysis based on approximating partial derivatives amounts to calculating Greeks, and can be used for calibration. Recall that for tree methods and for finite-difference methods there are easy ways to establish approximations to the Greeks delta, gamma, and theta, without the need for any recalculations. For Monte Carlo methods, this task is more costly. When results are required for slightly changed parameter values, to set up difference quotients, it may be necessary to rerun Monte Carlo.

As an example we comment on approximating delta$= \frac{\partial V}{\partial S}$. A simple approach is to apply two runs of Monte Carlo simulation, one for S_0 and one for a close value $S_0 - \Delta S$. Then an approximation of delta is obtained by the difference quotient

$$\frac{V(S_0) - V(S_0 - \Delta S)}{\Delta S}\,. \tag{3.35}$$

The increment ΔS must be chosen carefully and not too small, because (B1.6) in Appendix B1 tells us that the variance of (3.35) for arbitrary numerator scales with $(\Delta S)^{-2}$. So it is important to investigate how the numerator depends on ΔS. Simulating the two terms $V(S_0)$ and $V(S_0 - \Delta S)$ using common random numbers improves the situation, see [Gla04]. Computing time can be saved by working with series of precalculated random numbers. The crude approach symbolized by (3.35) does not require additional programming, but the costs are prohibitive for multiasset options.

With some more sophistication, the effort can be reduced. For example, options are often priced for different maturities. When Monte Carlo is combined with a bridging technique, several such options can be priced effectively in a single run [RiW03]. A general reference on estimating sensitivities is Chapter 7 in [Gla04].

There are alternatives improving accuracy and saving computing time. For example, Malliavin calculus allows to shift the differencing to the density function, which leads via a kind of integration by parts to a different integral to be approximated by Monte Carlo. For references on this technique consult [FoLLLT99].

Another method that speeds up a sensitivity analysis significantly is the adjoint method developed by [GiG06], which is described next.

Pathwise Sensitivities

Sensitivities can be approximated in a pathwise fashion. Consider similar as in (1.41) a system of autonomous SDEs

$$\mathrm{d}X_t = a(X_t)\,\mathrm{d}t + b(X_t)\,\mathrm{d}W_t \tag{3.36}$$

where $X_t \in \mathbb{R}^n$, and $W_t \in \mathbb{R}^m$ is a vector of independent Wiener processes. That is, b is $n \times m$ and takes care of possible correlations ($\longrightarrow$ Exercise 3.14).

For a standard discretization with M steps assume $t_0 = 0$, $T = \Delta t \cdot M$, $t_j := j\Delta t$, $j = 0, \ldots, M$, and let $\Psi(X(T))$ denote the *discounted* payoff. The Euler discretization of (3.36) is

$$y(t_{j+1}) = y(t_j) + a(y(t_j))\Delta t + b(y(t_j))Z(t_j)\sqrt{\Delta t}\,. \tag{3.37}$$

We consider one calculated path X_t, $0 \le t \le T$, represented by $y(t_j)$, $0 \le j \le M$, and keep its random vectors $Z(t_j)$ available. The aim is to estimate the sensitivity vector

$$s(0)^{tr} := \frac{\partial \Psi(X(T))}{\partial X(0)}$$

(taken as a row vector). By the chain rule,

$$s(0)^{tr} = \frac{\partial \Psi(X(T))}{\partial X(T)} \frac{\partial X(T)}{\partial X(0)}\,. \tag{3.38}$$

The first factor is easily available. The endeavor is to approximate the matrix $\frac{\partial X(T)}{\partial X(0)}$. To this end, we use the dynamics as created by the Euler method (3.37), and calculate the approximation

$$\frac{\partial y(T)}{\partial y(0)}\,.$$

As outlined in [Gla04, Section 7.2], we differentiate the ith component of the Euler formula (3.37) with respect to $y_k(t_0)$, which gives

$$\begin{aligned}\frac{\partial y_i(t_{j+1})}{\partial y_k(t_0)} =& \frac{\partial y_i(t_j)}{\partial y_k(t_0)} + \sum_{l=1}^{n} \frac{\partial a_i(y(t_j))}{\partial y_l(t_j)} \frac{\partial y_l(t_j)}{\partial y_k(t_0)} \Delta t \\ &+ \frac{\partial}{\partial y_k(t_0)} \sum_{\nu=1}^{m} b_{i\nu}(y(t_j)) Z_\nu(t_j) \sqrt{\Delta t}\end{aligned}$$

for all $i, k = 1, \ldots, n$. The last term is

$$\sum_{\nu=1}^{m} \sum_{l=1}^{n} \frac{\partial b_{i\nu}(y(t_j))}{\partial y_l(t_j)} \frac{\partial y_l(t_j)}{\partial y_k(t_0)} Z_\nu(t_j) \sqrt{\Delta t}\,.$$

With

$$\Delta_{ik}(j) := \frac{\partial y_i(t_j)}{\partial y_k(t_0)}$$

this is written

$$\begin{aligned}\Delta_{ik}(j+1) = \Delta_{ik}(j) &+ \sum_{l=1}^{n} \frac{\partial a_i(y(t_j))}{\partial y_l(t_j)} \Delta_{lk}(j) \Delta t \\ &+ \sum_{\nu=1}^{m} \sum_{l=1}^{n} \frac{\partial b_{i\nu}(y(t_j))}{\partial y_l(t_j)} \Delta_{lk}(j) Z_\nu(t_j) \sqrt{\Delta t}\,.\end{aligned} \tag{3.39}$$

This recursion (3.39) can be written in matrix notation. To this end, we use the definition (as in [GiG06]) of the entries of $(n \times n)$-matrices $D(j)$

$$D_{ik}(j) := \delta_{ik} + \frac{\partial a_i(y(t_j))}{\partial y_k(t_j)} \Delta t + \sum_{\nu=1}^{m} \frac{\partial b_{i\nu}(y(t_j))}{\partial y_k(t_j)} Z_\nu(t_j) \sqrt{\Delta t}\,. \tag{3.40}$$

(Here $\delta_{ik} = 1$ for $k = i$, and $= 0$ for $k \neq i$, is the Kronecker symbol and no dividend yield.) The resulting recursion for the $(n \times n)$-matrices $\Delta(j)$ with elements $\Delta_{ik}(j)$ is

$$\Delta(j+1) = D(j)\Delta(j), \quad j = 0, \ldots, M-1, \quad \Delta(0) = I\,. \tag{3.41}$$

This summarizes the evolution of the path in a *forward fashion*. After M matrix products the final matrix $\Delta(M)$ is the estimate $\frac{\partial y(T)}{\partial y(0)}$ for $\frac{\partial X(T)}{\partial X(0)}$. Then an approximation $\bar{s}(0)^{\text{tr}}$ of the sensitivity vector $s(0)^{\text{tr}}$ is obtained via the product (3.38).

Adjoint Method

As suggested by [GiG06], a backward view is possible too. To see this, rewrite the above as

$$\begin{aligned} \bar{s}(0)^{\text{tr}} :&= \frac{\partial \Psi(y(T))}{\partial y(T)} \frac{\partial y(T)}{\partial y(0)} = \frac{\partial \Psi(y(T))}{\partial y(T)} \Delta(M) \\ &= \frac{\partial \Psi(y(T))}{\partial y(T)} D(M-1) \cdot \ldots \cdot D(0) \end{aligned}\,.$$

The observation of [GiG06] is that $\bar{s}(0)$ can be calculated with a backward recursion, which operates n-vectors rather than $(n \times n)$-matrices. We start with the row vector

$$\bar{s}(M)^{\text{tr}} := \frac{\partial \Psi(y(T))}{\partial y(T)}$$

and obtain

$$(\bar{s}(M-1))^{\text{tr}} = \frac{\partial \Psi}{\partial y(T)} D(M-1)\,,$$

or

$$\bar{s}(M-1) = (D(M-1))^{\text{tr}}\, \bar{s}(M)\,.$$

The next row vector is

$$(\bar{s}(M-2))^{\text{tr}} = \frac{\partial \Psi}{\partial y(T)} D(M-1)D(M-2) = (\bar{s}(M-1))^{\text{tr}} D(M-2),$$

or

$$\bar{s}(M-2) = (D(M-2))^{\text{tr}}\, \bar{s}(M-1)\,,$$

and so on, which results in the recursion

$$\bar{s}(j) = (D(j))^{\text{tr}} \bar{s}(j+1), \quad j = M-1, \ldots, 0, \quad \bar{s}(M) = \left(\frac{\partial \Psi}{\partial y(T)} \right)^{\text{tr}}. \tag{3.42}$$

This backward recursion updates the n components of the vector s for every j, whereas the forward recursion (3.41) updates the n^2 entries of Δ in each step. Hence the forward recursion (3.41) involves a factor of n more arithmetic operations than the backward recursion. Consequently, the backward recursion should be significantly faster for $n > 1$. But there is one drawback of the potentially fast backward recursion: Its implementation requires to store the entire path of the y-vectors with their Z-vectors in order to have the D-matrices available. For very small step sizes Δt (M large) this deteriorates the speed somewhat. And switching to another payoff Ψ requires to recalculate the backward recursion, whereas the forward recursion can use the previous $\Delta(M)$ again. Observing these two features, the backward recursion (3.42) ("adjoint method") is highly advantageous. — The above method approximates pathwise deltas. In a similar way, sensitivities with respect to parameters can be calculated, see [GiG06].

Example 3.14 (Heston-Hull-White model)

Extending Heston's model (1.43) by an SDE for the interest rate r_t leads to the system

$$\begin{aligned} \mathrm{d}S_t &= r_t S_t \,\mathrm{d}t + \sqrt{v_t}\, S_t \,\mathrm{d}\tilde{W}_t^{(1)} \\ \mathrm{d}v_t &= \kappa(\theta - v_t)\,\mathrm{d}t + \sigma_2\sqrt{v_t}\,\mathrm{d}\tilde{W}_t^{(2)} \\ \mathrm{d}r_t &= \alpha(R(t) - r_t)\,\mathrm{d}t + \sigma_3\,\mathrm{d}\tilde{W}_t^{(3)} \end{aligned} \tag{3.43}$$

The function R in the mean-reversion term for r_t can be chosen as to match the current term structure [HaH10], here chosen as constant for simplicity:

$$R \equiv 0.06,\ \alpha = 0.1,\ \kappa = 3,\ \theta = 0.12,$$
$$\sigma_2 = 0.04,\ \sigma_3 = 0.01,\ T = 1,\ K = 100.$$

The mean reversion level $\theta = 0.12$ corresponds to a volatility of about 35%. The Brownian motions $\tilde{W}_t^{(1)}$, $\tilde{W}_t^{(2)}$, $\tilde{W}_t^{(3)}$ are assumed (partly) correlated:

$$\rho_{12} = 0.6,\ \rho_{13} = \rho_{23} = 0\,,$$

hence $\tilde{W}_t^{(3)}$ is not correlated with $\tilde{W}_t^{(1)}$, $\tilde{W}_t^{(2)}$. Accordingly, the Cholesky decomposition (Section 2.3.3) has a block structure, and Exercise 2.9 can be applied. To cast it into the framework of (1.41), observe $n = 3$,

$$X := \begin{pmatrix} S \\ v \\ r \end{pmatrix}, \quad a(X) = \begin{pmatrix} X_1 X_3 \\ \kappa(\theta - X_2) \\ \alpha(R - X_3) \end{pmatrix}$$

and

$$b(X)\,\mathrm{d}W_t = \begin{pmatrix} X_1\sqrt{X_2} & 0 & 0 \\ \sigma_2\sqrt{X_2}\,\rho_{12} & \sigma_2\sqrt{X_2}\sqrt{1-\rho_{12}^2} & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix} \begin{pmatrix} \mathrm{d}W_t^{(1)} \\ \mathrm{d}W_t^{(2)} \\ \mathrm{d}W_t^{(3)} \end{pmatrix}$$

with independent Wiener processes $W^{(i)}$. In the discretization the Wiener process can be taken as

$$\sqrt{\Delta t}\, Z_1(t),\ \sqrt{\Delta t}\, Z_2(t),\ \sqrt{\Delta t}\, Z_3(t)$$

with $Z_i \sim \mathcal{N}(0,1)$. $\sqrt{\Delta t}\, b(X)Z$ is a vector, an its partial derivatives enter (3.40).

For a concrete example, we price a European call. Since the interest rate is variable, we discount each trajectory with its proper rate. Hence, the discounted payoff is

$$\exp\left(-\int_0^T r_t \,\mathrm{d}t\right)(S_T - K)^+ .$$

For experiments, we have chosen the starting point

$$S_0 = 95,\ v_0 = \theta,\ r_0 = R\,,$$

approximated the discounting integral by the trapezoidal sum (C1.2), and obtained $V(S_0, v_0, r_0, 0) \approx 13.1$. The reader is encouraged to set up the matrix $D(j)$ and test the adjoint method.

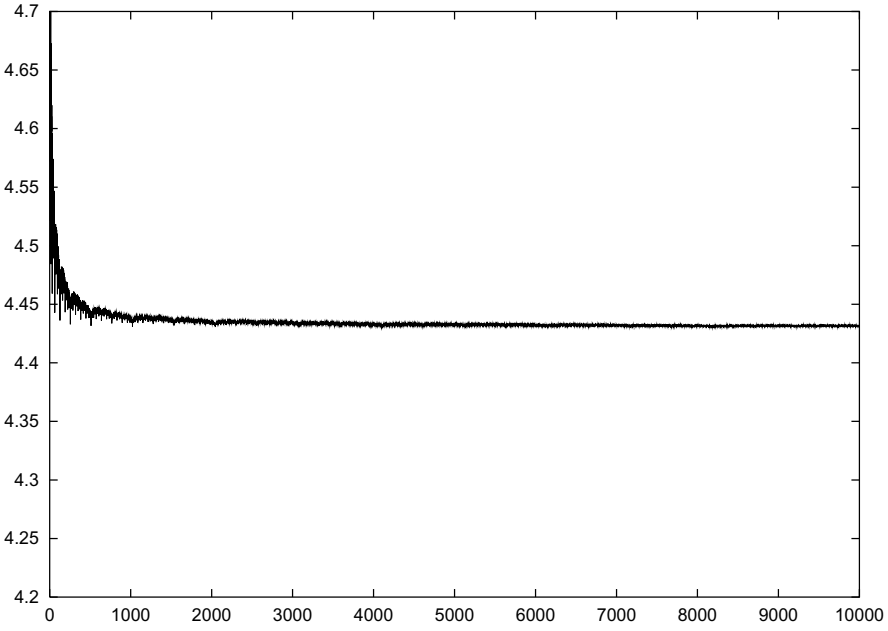

Fig. 3.15. Quasi Monte Carlo applied to Example 3.7

Test with Halton Points

To complete this chapter, we test the Monte Carlo simulation in a fully deterministic variant. To this end we insert the quasi-random two-dimensional Halton points into Algorithm 2.13 and use the resulting quasi normal deviates to calculate solutions of the SDE. In this way, for Example 3.7 acceptable accuracy is reached already for about 2000 paths, much better than what is shown in the experiments reported by Figures 3.3 or 3.5.

A closer investigation reveals that normal deviates based on Box-Muller-Marsaglia (Algorithm 2.13) with two-dimensional Halton points lose the equidistributedness; the low discrepancy is not preserved. Apparently the quasi-random method does not simulate independence [Gen98]. A related visual inspection resembles Figure 2.6. This sets the stage for the slightly faster inversion method [Moro95] ($\longrightarrow$ Appendix D2), based on one-dimensional low-discrepancy sequences. Figure 3.15 shows the result. The scaling of the figure is the same as before.

Notes and Comments

on Sections 3.1, 3.2:

Under suitable assumptions it is possible to prove existence and uniqueness for strong solutions, see [KlP92]. Usually the discretization error dominates other sources of error. We have neglected the sampling error (the difference between $\widehat{\epsilon}$ and ϵ), imperfections in the random number generator, and rounding errors. Typically these errors are likely to be less significant. Section 3.2 closely follows Section 5.1 of [KlP92].

on Section 3.3:

[KlP92] discusses many methods for the approximation of paths of SDEs, and proves their convergence. An introduction is given in [Pla99]. Possible orders of strongly converging schemes are integer multiples of $\frac{1}{2}$ whereas the orders of weakly converging methods are whole numbers. Simple adaptions of deterministic schemes do not converge for SDEs. For the integration of *random ODEs* we refer to [GrK01]. Maple routines for SDEs can be found in [CyKO01], and MATLAB routines in [Hig01].

For ODEs and SDEs linear stability is investigated. This is concerned with the long-time behavior of solutions of the test equation $\mathrm{d}X_t = \alpha X_t\,\mathrm{d}t + \beta X_t\,\mathrm{d}W_t$, where α is a complex number with negative real part. This situation does not appear relevant for applications in finance. The numerical stability in the case $Re(\alpha) < 0$ depends on the step size h and the relation among the three parameters α, β, h. For this topic and further references we refer to [SaM96], [Hig01], [Pla99].

on Section 3.4:

For Brownian bridges see, for instance, [KaS91], [ReY91], [KlP92], [Øk98], [Mor98], [Gla04]. Other bridges than Brownian bridges are possible. For a Gamma process and a Gaussian bridge this is shown in [RiW02], [RiW03]. For the effectiveness of Monte Carlo integration improved with bridging techniques, see [CaMO97]. The probability that a Brownian bridge passes a given barrier is found in [KaS91], see also [Gla04]. The maximum of a Wiener process tied down to $W_0 = 0$, $W_1 = a$ on $0 \le t \le 1$ has the distribution $F(x)$ of Exercise 2.16. And the time instant at which the maximum is attained is distributed with

$$F(x) = \frac{2}{\pi} \arcsin(\sqrt{x}) \quad \text{for } 0 \le x \le 1\,.$$

Another alternative to fill large gaps is to apply fractal interpolation [Man99].

on Section 3.5:

In the literature the basic idea of the approach summarized by equation (3.19) is analyzed using martingale theory, compare the references in Chapter 1 and Appendix B2. An early paper suggesting MC for the pricing of options is [Boy77]. The calculation of risk indices such as *value at risk* is an important application of Monte Carlo methods, see the notes on Section 1.8. The equivalence of the Monte Carlo simulation (representation (3.18)/(3.19)) with the solution of the Black–Scholes equation is guaranteed by the theorem of Feynman and Kac [KaS91], [Nef96], [Reb96], [Øk98], [Bjö98], [TaR00], [Shr04]. A standard reference on MC in finance is [Gla04].

Monte Carlo simulations can be parallelized in a trivial way: The single simulations can be distributed among the processors in a straightforward fashion because they are independent of each other. If M processors are available, the speed reduces by a factor of $1/M$. But the streams of random numbers in each processor must be independent. For related generators see [Mas99]. In doubtful and sensitive cases Monte Carlo simulation should be repeated with other random-number generators, and with low-discrepancy numbers [Jäc02].

The method of control variates can be modified with a parameter α,

$$V_{\mathrm{CV}}^{\alpha} := \widehat{V} + \alpha(V^* - \widehat{V}^*),$$

where one tries to find a value of α such that the variance is minimized. For a discussion of variance reduction and examples, consult Chapter 4 in [Gla04]. For the variance-reduction method of *importance sampling*, see [New97], [Gla04]. In particular, a change of drift helps driving the underlying assets into "important" regions. An optimal drift is possible that reduces the variance significantly. [Aro03] suggests a truncated version of the Robbins-Monro algorithm, and [Jon11] reduces the number of insignificant paths for his robust regression with a deterministic method.

on Section 3.6:

For Monte Carlo simulation on American options see also [BrG97], [BoBG97], [Kwok98], [Rog00], [Fu01], [LonS01], [Gla04]. Note that for multivariate options of the American style the costs are increasing with the dimension more significantly than for European options. For parametric methods, the parameter vector β defines surfaces rather than curves. And for regression methods, the calculation of C or $\hat{C}$ is costly and does depend on the dimension. A nice experiment with a parametric method is given in [Hig04]. Significant savings are possible when the dimension is reduced by a principle component analysis ($\longrightarrow$ Exercise 2.18).

A first version of regression was introduced by [Til93], where the continuation value was approximated based on subsets of paths. This bundling technique was modified in [Car96] by an improved regression. As [Til93] points out, a single set of paths of an underlying asset can be generated and then used repeatedly to value many different derivatives. Lack of independence makes it difficult to prove convergence, or to set up confidence intervals. For these aspects, see [Egl05], and [AnB04] and the references therein.

Exercises

Exercise 3.1 Implementing Euler's Method

Implement Algorithm 1.11. Start with a test version for one scalar SDE, then develop a version for a system of SDEs. Test examples:

a) Perform the experiment of Figure 1.17.
b) Integrate the system of Example 1.15 for $\alpha = 0.3$, $\beta = 10$ and the initial values $S_0 = 50$, $\sigma_0 = 0.2$, $\xi_0 = 0.2$ for $0 \le t \le 1$.

We recommend to plot the calculated trajectories.

Exercise 3.2 Itô Integral in Equation (3.9)

Let the interval $0 \le s \le t$ be partitioned into n subintervals, $0 = t_1 < t_2 < \ldots < t_{n+1} = t$. For a Wiener process W_t assume $W_{t_1} = 0$.

a) Show $\displaystyle\sum_{j=1}^{n} W_{t_j}\left(W_{t_{j+1}} - W_{t_j}\right) = \frac{1}{2}W_t^2 - \frac{1}{2}\sum_{j=1}^{n}\left(W_{t_{j+1}} - W_{t_j}\right)^2$

b) Use Lemma 1.9 to deduce Equation (3.9).

Exercise 3.3 Integration by Parts for Itô Integrals

a) Show

$$\int_{t_0}^{t} s\,\mathrm{d}W_s = tW_t - t_0W_{t_0} - \int_{t_0}^{t} W_s\,\mathrm{d}s$$

Hint: Start with the Wiener process $X_t = W_t$ and apply the Itô Lemma with the transformation $y = g(x,t) := tx$.

b) Denote $\Delta Y := \int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}W_z \, \mathrm{d}s$. Show by using a) that

$$\int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}z \, \mathrm{d}W_s = \Delta W \Delta t - \Delta Y \, .$$

Exercise 3.4 Moments of Itô Integrals for Weak Solutions

a) Use the Itô isometry

$$\mathsf{E}\left[\left(\int_a^b f(t,\omega)\,\mathrm{d}W_t\right)^2\right] = \int_a^b \mathsf{E}\left[f^2(t,\omega)\right]\,\mathrm{d}t$$

to show its generalization

$$\mathsf{E}\left[I(f)I(g)\right] = \int_a^b \mathsf{E}[fg]\,\mathrm{d}t \, , \qquad \text{where} \quad I(f) = \int_a^b f(t,\omega)\,\mathrm{d}W_t \, .$$

Hint: $4fg = (f+g)^2 - (f-g)^2$.

b) For $\Delta Y := \int_{t_0}^{t} \int_{t_0}^{s} \mathrm{d}W_z \, \mathrm{d}s$ the moments are

$$\mathsf{E}[\Delta Y] = 0, \quad \mathsf{E}[\Delta Y^2] = \frac{\Delta t^3}{3}, \quad \mathsf{E}[\Delta Y \Delta W] = \frac{\Delta t^2}{2} \quad \text{and} \quad \mathsf{E}[\Delta Y \Delta W^2] = 0.$$

Show this by using a) and $\mathsf{E}\left[\int_a^b f(t,\omega)\,\mathrm{d}W_t\right] = 0$.

Exercise 3.5

By transformation of two independent standard normally distributed random variables $Z_i \sim \mathcal{N}(0,1)$, $i = 1,2$, two new random variables are obtained by

$$\Delta\widehat{W} := Z_1\sqrt{\Delta t}, \qquad \Delta\widehat{Y} := \frac{1}{2}(\Delta t)^{3/2}\left(Z_1 + \frac{1}{\sqrt{3}}Z_2\right) .$$

Show that $\Delta\widehat{W}$ and $\Delta\widehat{Y}$ have the moments of (3.14).

Exercise 3.6

In addition to (3.14) further moments are

$$\mathsf{E}(\Delta W) = \mathsf{E}(\Delta W^3) = \mathsf{E}(\Delta W^5) = 0, \quad \mathsf{E}(\Delta W^2) = \Delta t, \quad \mathsf{E}(\Delta W^4) = 3\Delta t^2.$$

Assume a new random variable $\Delta\widetilde{W}$ satisfying

$$\mathsf{P}\left(\Delta\widetilde{W} = \pm\sqrt{3\Delta t}\right) = \frac{1}{6}, \qquad \mathsf{P}\left(\Delta\widetilde{W} = 0\right) = \frac{2}{3}$$

and the additional random variable

$$\Delta\widetilde{Y} := \frac{1}{2}\Delta\widetilde{W}\Delta t\,.$$

Show that the random variables $\Delta\widetilde{W}$ and $\Delta\widetilde{Y}$ have up to terms of order $O(\Delta t^3)$ the same moments as ΔW and ΔY.

Exercise 3.7 Brownian Bridge
For a Wiener process W_t consider

$$X_t := W_t - \frac{t}{T}W_T \quad \text{for } 0 \le t \le T\,.$$

Calculate $\mathsf{Var}(X_t)$ and show that

$$\sqrt{t\left(1-\frac{t}{T}\right)}Z \quad \text{with } Z \sim \mathcal{N}(0,1)$$

is a realization of X_t.

Exercise 3.8 Error of the Milstein Scheme
To which formula does the Milstein scheme reduce for linear SDEs? Perform the experiment outlined in Example 3.2 using the Milstein scheme of Algorithm 3.5. Set up a table similar as in Table 3.1 to show

$$\widehat{\varepsilon}(h) \approx h$$

for Example 3.2.

Exercise 3.9 Monte Carlo and European Option
For a European put with time to maturity $\tau := T - t$ prove that

$$\begin{aligned} V(S_t,t) &= \mathrm{e}^{-r\tau}\int_0^\infty (K-S_T)^+ \frac{1}{S_T\sigma\sqrt{2\pi\tau}}\exp\left\{-\frac{[\ln(S_T/S_t)-(r-\frac{\sigma^2}{2})\tau]^2}{2\sigma^2\tau}\right\}\mathrm{d}S_T \\ &= \mathrm{e}^{-r\tau}KF(-d_2) - S_tF(-d_1)\,, \end{aligned}$$

where d_1 and d_2 are defined in (A4.10).

Hints: The second equation is to be shown, the first only collects the terms of (3.18). Use $(K-S_T)^+ = 0$ for $S_T > K$, and get two integrals.

Exercise 3.10 Bias of the Euler Approximation
Given is the SDE $\mathrm{d}S_t = S_t(\mu\,\mathrm{d}t + \sigma\,\mathrm{d}W_t)$ with constant μ, σ. Let $\hat{S}$ denote an Euler approximation at $t_2 := 2\Delta t$, calculated with two steps of length Δt, starting at $t_0 := 0$ with the value S_0.
a) Calculate $\mathsf{E}(\hat{S})$.
b) Calculate the bias $\mathsf{E}(\hat{S}) - S_0\exp[\mu t_2]$.

Exercise 3.11 Monte Carlo for European Options

Implement a Monte Carlo method for single-asset European options, based on the Black–Scholes model. Perform experiments with various values of N and a random number generator of your choice. Compare results obtained by using the analytic solution formula for S_t with results obtained by using Euler's discretization. For c) B is the barrier such that the option expires worthless when $S_t \geq B$ for some t.

input: S_0, number of simulations (trajectories) N, payoff function $\Psi(S)$, risk-neutral interest rate r, volatility σ, time to maturity T, strike K.

payoffs:

a) vanilla put, with $\Psi(S) = (K - S)^+$, $S_0 = 5$, $K = 10$, $r = 0.06$, $\sigma = 0.3$, $T = 1$.
b) binary call, with $\Psi(S) = \mathbf{1}_{S>K}$, $S_0 = K = \sigma = T = 0.5$, $r = 0.1$
c) up-and-out barrier: call with $S_0 = 5$, $K = 6$, $r = 0.05$, $\sigma = 0.3$, $T = 1$, $B = 8$.

Hint: Correct values are: a) 4.43046 b) 0.46220 [Que07] c) 0.0983 [Hig04]

Exercise 3.12 Project: Monte Carlo Experiment

Construct as hitting curve a parabola with horizontal tangent at $(S, t) = (K, T)$, similar as in Figure 3.10. The parabola is defined by the intersection with the S-axis, $(S, t) = (\beta, 0)$. Choose $K = 10$, $r = 0.006$, $\sigma = 0.3$, and $S_0 = 9$ and simulate for several values of β the GBM $\mathrm{d}S = rS\,\mathrm{d}t + \sigma S\,\mathrm{d}W$ several thousand times, and calculate the hitting time for each trajectory. Estimate a lower bound to $V(S_0, 0)$ using (3.30). Decide whether an exact calculation of the hitting point makes sense. (Run experiments comparing such a strategy to implementing the hitting time restricted to the discrete time grid.) Think about how to implement upper bounds.

Exercise 3.13 Error of Biased Monte Carlo

Assume

$$\mathrm{MSE} = \zeta(h, N) := \alpha_1^2 h^{2\beta} + \frac{\alpha_2}{N}$$

as error model of a Monte Carlo simulation with sample size N, based on a discretization of an SDE with stepsize h, where α_1, α_2 are two constants.

a) Argue why for some constant α_3

$$C(h, N) := \alpha_3 \frac{N}{h}$$

is a reasonable model for the costs of the MC simulation.

b) Minimize $\zeta(h, N)$ with respect to h, N subject to the side condition

$$\alpha_3 N/h = C$$

for given budget C.

c) Show that for the optimal h, N

$$\sqrt{\text{MSE}} = \alpha_4 C^{-\frac{\beta}{1+2\beta}} .$$

Exercise 3.14 SDE in Standard Form

Let us denote (1.41) as "standard form" of a system of SDEs, with uncorrelated Wiener processes $W_t^{(1)}, \ldots, W_t^{(m)}$. What is the vector a and the matrix b for

a) the example of equation (3.28),
b) the Heston model of equation (1.43).

For the Heston model, first transform the unknown v_0 to the right-hand side by scaling $\tilde{v}_t := v_t / v_0$.

Exercise 3.15 Binary Random Variate

Let α, β, p with $0 < p < 1$ be given numbers. Design an algorithm that outputs α with probability p and β with probability $1 - p$.