10

Numerical Solutions Using Finite Differences

One of the best known and widely used numerical methods to solve partial differential equations in finance and elsewhere is the *finite difference method*. Finite difference methods are very powerful and flexible as well. They can be applied to a wide variety of various derivatives. Different exercise modes, including European (exercise at expiry only), American (exercise at any time) or Bermudan (exercise at a limited set of exercise dates), could be implemented without major problems. In comparison with the tree methods introduced in Chap. 9, the finite difference method possesses superior convergence features, which justifies the greater initial effort for its implementation. Therefore, we will provide a very detailed discussion of this important method.

Finite difference methods approximate the partial derivatives appearing in partial differential equations like the Black-Scholes equation 7.8 using finite difference quotients. The equation is then solved on a *grid* spanned by the linearly independent variables (for example, time t and price S of the underlying) appearing in the PDE.[1] Doing so, we obtain a solution surface which represents the price on each of the grid points (S, t). In general, a PDE has an unbounded number of solutions. Usually we are only interested in a solution which satisfies specific boundary and/or initial conditions. The finite difference method requires the specification of both boundary and initial conditions.

[1] In the following we will often use the abbreviation *PDE* for "partial differential equation", as is common practice in the related literature.

H.-P. Deutsch, M. W. Beinker, *Derivatives and Internal Models*, Finance and Capital Markets Series, https://doi.org/10.1007/978-3-030-22899-6_10

Since finite differences can be applied in quite general settings, we will require only the assumptions needed for arbitrage free trading, i.e., Assumptions 1, 2, 3,5 from Chap. 4, with the additional Assumption 6. Assumption 6 ensures that the variables in the problem are continuous. This is necessary if we want to obtain differentiable solutions. In order to provide a manageable overview of finite differences, we will restrict the treatment in the examples given here to random walks (Assumption 7) with non-stochastic interest rates and volatilities (Assumptions 8 and 10, respectively). In addition, we will not consider counter party default risk (Assumption 4).

10.1 Discretizing the Black-Scholes Equation

Below, the Black-Scholes Differential Eq. 7.8 will be solved numerically with the help of finite difference methods. In the literature, Eq. 7.8 is often transformed into an equation based on a new variable given by $Z = \ln(S)$. Such an equation has the advantage that the coefficients no longer depend explicitly on S. On occasion, it is claimed that a further advantage of this change of variable is that a uniform grid in Z is numerically more efficient than a uniform grid in S. However, the differences are usually negligible and this argument does not hold for barrier options. Furthermore, it is often preferable to use a non-uniform grid. We could distribute the grid points logarithmically, for example so that the non-uniform grid in S corresponds to a uniform grid in Z. In view of these considerations, we will continue to use the Black-Scholes equation in the form given by 7.8. But we will present the finite difference method for general *non-uniform grids*.

The finite difference method now consists in determining the value V of a financial instrument on a grid with coordinates S and t by approximating the partial derivatives with *finite* differences. We will restrict the discussion here to a rectangular grid allowing, however, the distance between grid points to be non-uniform. Such a grid is completely determined by the grid points in the S and t directions denoted by:

$$\begin{aligned} S_i &; \; i = 0, 1, 2, \ldots, M \\ t_j &; \; j = 0, 1, 2, \ldots, N \end{aligned} \tag{10.1}$$

We introduce the notation $W_{i,j} = W(S_i, t_j)$ for the solution's approximation in order to distinguish it from the exact solution $V(S_i, t_j)$ evaluated at points on the grid.

For the sake of clarity we will at several instances start with a grid whose time steps as well as the steps in the direction of the underlying price are uniformly spaced. For such *uniform grids* the spacing between grid points is simply

$$S_i - S_{i-1} = \delta S \; \forall i \implies S_i = S_0 + i\,\delta S \tag{10.2}$$
$$t_j - t_{j-1} = \delta t \; \forall j \implies t_i = t_0 + i\,\delta t \; .$$

We will then, however, always generalize our discussion to non-uniform grids.

The fundamental idea behind the method presented here is to determine the value of the derivative from the values at neighboring time points. Since the value of the derivative (as a function of the underlying's price) is usually known at maturity (it is given a priori by the payoff profile $P(S)$ as a function of S), we proceed using the strategy of starting at maturity $t_N = T$ and calculating *backwards* to time t_{N-1}, from there calculating back to t_{N-2} and so on. To accomplish this, we express the value of the instrument in terms of its Taylor series with respect to time, and express the time derivatives appearing in the Taylor expansion in terms of the derivatives with respect to the underlying price by using the Black-Scholes PDE. Here, the similarity to *backward induction* as described in Sect. 14.6.1 is not accidental.

Using the Black Scholes equation 7.8, the partial derivative of V with respect to time is expressed in terms of derivatives with respecto to S:

$$\frac{\partial V(S,t)}{\partial t} = r(t)V(S,t) \; - [r(t) - q(t)]\, S\frac{\partial V(S,t)}{\partial S} - \frac{1}{2}\sigma^2(S,t)S^2\frac{\partial^2 V(S,t)}{\partial S^2} \; . \tag{10.3}$$

The partial derivatives with respect to S in this equation are approximated by quotients of finite differences, as will be explicitly demonstrated in the next sections.

10.1.1 The Explicit Method

In the explicit method, the Taylor series expansion is used to calculate values at an *earlier* time $t - \delta t$ from the values at time t:

$$V(S, t - \delta t) = V(S,t) - \delta t \frac{\partial V(S,t)}{\partial t} + O(\delta t^2) \; .$$

The last term on the right-hand side of the equation states that no terms will be considered which are of order two or greater with respect to the time difference δt. If the time difference is small enough, we can assume that the time dependence of $V(S, t)$ can be adequately described if we simply neglect terms of order $O(\delta t^2)$. Note that on the left-hand side of the above equation, the derivative of the value evaluated at time $t - \delta t$ appears, whereas the right-hand side consists of terms evaluated at time t. The trick is now *not* to express the right-hand side in terms of a difference quotient in time (we would not have accomplished anything by doing so since it would involve introducing another time point $t - \delta t$ or $t + \delta t$) but rather to express the time derivative in terms of partial derivatives with respect to S obtained from the Black-Scholes equation 7.8. These partial derivatives with respect to S are evaluated at time t so that the value of the derivative at an earlier time $t - \delta t$ can in fact be recovered solely from information available at time t. Here, we *explicitly* calculate earlier values from those (known) values from a later time. This method is thus referred to as the *explicit method.*

10.1.2 The Implicit Method

In the implicit method, we use the Taylor series expansion in the time variable to obtain an expression for values at a *later* time $t + \delta t$ from the values at time t:

$$V(S, t + \delta t) = V(S, t) + \delta t \frac{\partial V(S, t)}{\partial t} + O(\delta t^2) \; .$$

In this case, later (known) values are expanded in terms of earlier (unknown) ones. This expansion can only be used *implicitly* to calculate the unknown values from the known ones, hence the name *implicit method*.

10.1.3 Combinations of Explicit and Implicit Methods (Crank-Nicolson)

The two methods described above can be combined by taking a linear combination of the two respective Taylor series expansions. To avoid the appearance of three different time points in the resulting expression ($t - \delta t$, t and $t + \delta t$) a change in variable in one of the Taylor series should be made.

For example, the transformation $t \to t + \delta t$ in the Taylor series expansion for the explicit method (and dividing by δt) yields an equation of the form:

$$\frac{V(S, t+\delta t) - V(S,t)}{\delta t} = \frac{\partial V(S, t+\delta t)}{\partial t} + O(\delta t) \; . \tag{10.4}$$

Here, we have rearranged the Taylor series so that the left-hand side is written in terms of a *difference quotient* in t, while on the right-hand side a *differential* quotient appears (which will later be replaced by a difference quotient with respect to the underlying price). Note that dividing through by δt has the effect of reducing the order of the error term to a linear order in δt. An analogous procedure for the implicit method allows its respective Taylor series to be rearranged as well (here, a variable transformation is unnecessary since the series is already expressed in terms of t and $t + \delta t$):

$$\frac{V(S, t+\delta t) - V(S,t)}{\delta t} = \frac{\partial V(S, t)}{\partial t} + O(\delta t) \; . \tag{10.5}$$

The only thing now distinguishing the two expressions is that the differential quotient on the right-hand side (and thus, the difference quotients with respect to S yet to be determined) is written in terms of the time t, whereas in the expression derived from the explicit method, the differential quotient is expressed in terms of the later time point $t + \delta t$. Naturally, the equality holds if we take any linear combination of the two equations:

$$\frac{V(S, t+\delta t) - V(S,t)}{\delta t} = (1-\theta)\frac{\partial V(S, t+\delta t)}{\partial t} + \theta \frac{\partial V(S, t)}{\partial t} + O(\delta t) \, , \, 0 \le \theta \le 1 \; .$$

Note the following correspondence[2] between the notation above and the discrete notation introduced in Eq. 10.1:

$$\begin{aligned} S \mathrel{\widehat{=}} S_i \quad , \quad t \mathrel{\widehat{=}} t_j \quad , \quad t + \delta t \mathrel{\widehat{=}} t_{j+1} \\ W_{i,j} = W(S_i, t_j) \mathrel{\widehat{=}} V(S,t) \\ W_{i,j+1} = W(S_i, t_{j+1}) \mathrel{\widehat{=}} V(S, t+\delta t) \; . \end{aligned}$$

[2] The sign $\widehat{=}$ means "corresponds to".

In the discrete notation the above equation reads

$$\frac{W_{i,j+1} - W_{i,j}}{t_{j+1} - t_j} = (1-\theta)\frac{\partial V(S_i, t_{j+1})}{\partial t} + \theta\frac{\partial V(S_i, t_j)}{\partial t} + O\left(t_{j+1} - t_j\right) \ , 0 \le \theta \le 1 \ . \tag{10.6}$$

In this notation, the equation holds for *non*-uniform grids, as well. Setting $\theta = 1$, we obtain the implicit method, with $\theta = 0$ the explicit method. The particular choice of $\theta = 1/2$ has a special name. It is known as the *Crank-Nicolson method*.

Equations 10.4 and 10.5 can also be interpreted as follows: the differen*ce* quotients ("finite differences") on the left-hand side of the equations are approximations of the differen*tial* quotients with respect to time found on the right-hand side of the equations.

The above approximations are exact up to *linear* terms in δt. There are several methods available for approximating partial derivatives using finite differences. A greater accuracy can be obtained if, for instance, all three time points t_{j-1}, t_j and t_{j+1} are included in the finite differences approximating the time derivative. Then the approximation is exact up to second order. This three-time procedure requires that the time derivative be approximated by a carefully selected convex combination of forward, backward and symmetric finite differences. Another possibility is to use symmetric finite differences. This also gives an approximation exact up to second order, however, does not lead to a stable procedure for solving the differential equation. In this book we will not pursue such more precise approximations for the time derivative.

10.1.4 Symmetric Finite Differences of the Underlying Price

For the sake of consistency, the difference quotients with respect to the underlying price S should be exact up to order $O(\delta S^2)$ since it follows from the random walk assumption that $dS \sim \sqrt{dt}$ and thus

$$\delta t \sim \delta S^2 \ .$$

This means that in order to attain the same degree of accuracy as in the time direction, the approximation in the S direction must be exact up to order $O(\delta S^2)$. To achieve this we will use *symmetric differences* to approximate the

derivatives of first and second order with respect to the underlying needed for Eq. 10.3.

In order to demonstrate the concepts without making the notation unnecessarily complicated, we begin by assuming that the distance between grid points is constant as in Eq. 10.2 and then generalize to non-uniform grids. As above, we expand the value function in its Taylor series, this time in the S dimension:

$$\begin{aligned} V(S-\delta S,t) &= V(S,t) - \delta S\frac{\partial V(S,t)}{\partial S} + \frac{1}{2}\delta S^2\frac{\partial^2 V(S,t)}{\partial S^2} \\ &\quad - \frac{1}{6}\delta S^3\frac{\partial^3 V(S,t)}{\partial S^3} + O(\delta S^4) \\ V(S+\delta S,t) &= V(S,t) + \delta S\frac{\partial V(S,t)}{\partial S} + \frac{1}{2}\delta S^2\frac{\partial^2 V(S,t)}{\partial S^2} \\ &\quad + \frac{1}{6}\delta S^3\frac{\partial^3 V(S,t)}{\partial S^3} + O(\delta S^4)\ . \end{aligned}$$

Subtracting the first equation from the second and subsequently dividing by δS yields an approximation of the first derivative which is exact up to second order $O(\delta S^2)$. Adding the two equations and then dividing by δS^2 yields an approximation of the second derivative which is also exact up to second order $O(\delta S^2)$:

$$\begin{aligned} \frac{\partial V(S,t)}{\partial S} &= \frac{V(S+\delta S,t) - V(S-\delta S,t)}{2\,\delta S} + O(\delta S^2) \qquad (10.7) \\ \frac{\partial^2 V(S,t)}{\partial S^2} &= \frac{V(S+\delta S,t) - 2\,V(S,t) + V(S-\delta S,t)}{\delta S^2} + O(\delta S^2)\ . \end{aligned}$$

For general, *non*-uniforms grids, the above expressions are somewhat more complicated and their derivation is a bit more technical. The principle, however, remains the same: the partial derivatives of V with respect to the underlying evaluated at point S_i can be approximated up to order two with symmetric differences. To this end, V evaluated at $S_{i-1}$and S_{i+1}, is first expanded in its Taylor series about the points S_i:

$$\begin{aligned} V(S_{i-1},t) &= V(S_i,t) - \frac{\partial V(S_i,t)}{\partial S}(S_i - S_{i-1}) \\ &\quad + \frac{1}{2}\frac{\partial^2 V(S_i,t)}{\partial S^2}(S_i - S_{i-1})^2 + O\left((S_i - S_{i-1})^3\right) \end{aligned}$$

$$V(S_{i+1},t) = V(S_i,t) + \frac{\partial V(S_i,t)}{\partial S}(S_{i+1}-S_i) + \frac{1}{2}\frac{\partial^2 V(S_i,t)}{\partial S^2}(S_{i+1}-S_i)^2 + O\left((S_{i+1}-S_i)^3\right) .$$

We need this approximation for the partial derivatives to be exact up to second order. We thus neglect all terms of order $O((\delta S_{\max})^3)$ where $\delta S_{\max}$ denotes the greatest distance between two neighboring nodes in the S-grid. Unlike the uniform grid case, mere addition and subtraction of the two equations does not isolate the desired differential quotients since is possible that $(S_i - S_{i-1}) \neq (S_{i+1} - S_i)$. To overcome this inconvenience, we attempt to express the differential quotients as a linear combination of the function evaluated at the points S_i, $S_{i-1}$and S_{i+1}. We begin by assuming that the first derivative can be written as

$$\frac{\partial V(S_i,t)}{\partial S} = aV(S_{i-1},t) + bV(S_i,t) + cV(S_{i+1},t) .$$

Substituting the above Taylor series for $V(S_{i+1},t)$ and $V(S_{i-1},t)$ into this equation and using the linear independence of $V(S_i,t)$, $\frac{\partial V(S_i,t)}{\partial S}$ and $\frac{\partial^2 V(S_i,t)}{\partial S^2}$ to compare their coefficients leads to the following system of equations for the unknown coefficients a, b and c

$$\begin{aligned} a + b + c &= 0 \\ -(S_i - S_{i-1})a + (S_{i+1} - S_i)c &= 1 \\ (S_i - S_{i-1})^2 a + (S_{i+1} - S_i)^2 c &= 0 \end{aligned}$$

which has the solution

$$\begin{aligned} a &= -\frac{S_{i+1} - S_i}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} \\ b &= \left(\frac{S_{i+1} - S_i}{S_i - S_{i-1}} - \frac{S_i - S_{i-1}}{S_{i+1} - S_i}\right)\frac{1}{S_{i+1} - S_{i-1}} \\ c &= +\frac{S_i - S_{i-1}}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} \end{aligned}$$

The desired approximation of the first partial derivative with respect to S is thus

$$\frac{\partial V(S_i,t)}{\partial S} = \frac{1}{S_{i+1}-S_{i-1}}\left[-\frac{S_{i+1}-S_i}{S_i-S_{i-1}}V(S_{i-1},t)\right. \tag{10.8}$$
$$\left.+\left(\frac{S_{i+1}-S_i}{S_i-S_{i-1}}-\frac{S_i-S_{i-1}}{S_{i+1}-S_i}\right)V(S_i,t)+\frac{S_i-S_{i-1}}{S_{i+1}-S_i}V(S_{i+1},t)\right]$$
$$+O\left((\delta S_{\max})^2\right)$$

For the special case of a uniform S-grid with $S_{i+1}-S_i = S_i - S_{i-1} = \delta S$, this reduces to the expression in Eq. 10.7.

The same approach can be taken to isolate the second derivative. This time, we assume that the second derivative can be represented as a linear combination of the value function evaluated at the points S_{i-1}, S_i and S_{i+1}:

$$\frac{\partial^2 V(S_i,t)}{\partial S^2} = aV(S_{i-1},t)+bV(S_i,t)+cV(S_{i+1},t)\,.$$

As was done above, the Taylor expansions for $V(S_{i+1},t)$ and $V(S_{i-1},t)$ are substituted into this equation. Comparing the coefficients of $V(S_i,t)$, $\frac{\partial V(S_i,t)}{\partial S}$ and $\frac{\partial^2 V(S_i,t)}{\partial S^2}$ again leads to a system of equations for the unknown coefficients a, b and c, now given by

$$\begin{aligned} a+b+c &= 0\\ -(S_i-S_{i-1})a+(S_{i+1}-S_i)c &= 0\\ (S_i-S_{i-1})^2a+(S_{i+1}-S_i)^2c &= 2 \end{aligned}$$

which has the solution

$$\begin{aligned} a &= \frac{2}{(S_i-S_{i-1})(S_{i+1}-S_{i-1})}\\ b &= -\frac{2}{S_{i+1}-S_{i-1}}\left(\frac{1}{S_i-S_{i-1}}+\frac{1}{S_{i+1}-S_i}\right)\\ c &= \frac{2}{(S_{i+1}-S_i)(S_{i+1}-S_{i-1})}\,. \end{aligned}$$

The approximation of the second derivative of V with respect to S is now given by

$$\frac{\partial^2 V(S_i,t)}{\partial S^2} = \frac{2}{S_{i+1}-S_{i-1}}\left[\frac{V(S_{i-1},t)}{S_i - S_{i-1}} - \left(\frac{1}{S_i - S_{i-1}} + \frac{1}{S_{i+1}-S_i}\right) V(S_i,t) + \frac{V(S_{i+1},t)}{S_{i+1}-S_i}\right] + O(\delta S_{\max}) \; . \tag{10.9}$$

For a uniform S-grid this reduces to the expression in Eq. 10.7.

10.2 Difference Schemes

The approximations for the derivatives with respect to S can now be substituted into Eq. 10.3. For the sake of simplicity, we again consider first the case of a uniform grid as in Eq. 10.2, i.e., we will use the approximation given by Eq. 10.7. Under this assumption $S_i = S_0 + i\,\delta S$ holds and the approximation 10.3 for the differential quotient with respect to t evaluated at the point $t = t_j$ takes the form

$$\begin{aligned}\frac{\partial V(S_i,t_j)}{\partial t} &\approx r_j W_{i,j} - (r_j - q_j)\, S_i \, \frac{W_{i+1,j} - W_{i-1,j}}{2\delta S} \\ &\quad - \frac{1}{2}\,\sigma_{i,j}^2\, S_i^2\, \frac{W_{i+1,j} - 2\,W_{i,j} + W_{i-1,j}}{\delta\, S^2} \\ &= A_{i,j}\, W_{i-1,j} + B_{i,j}\, W_{i,j} + C_{i,j}\, W_{i+1,j}\end{aligned}$$

where

$$\begin{aligned} A_{i,j} &= \frac{S_i}{2\delta S}\left(r_j - q_j - \frac{S_i}{\delta S}\,\sigma_{i,j}^2\right) \\ B_{i,j} &= r_j + \left(\frac{S_i}{\delta S}\right)^2 \sigma_{i,j}^2 \\ C_{i,j} &= -\frac{S_i}{2\delta S}\left(r_j - q_j + \frac{S_i}{\delta S}\sigma_{i,j}^2\right) \; . \end{aligned} \tag{10.10}$$

On the right-hand side of the equation, we have again denoted the approximation of the exact solution $V(S_i, t_j)$ evaluated at the grid points by $W_{i,j} = W(S_i, t_j)$. At this stage, we allow the interest rate and the volatility to depend on time. This is the only reason why the coefficients A, B and C have been equipped with the index j. For time-independent interest rates and volatilities, the index j on the coefficients is superfluous.

The corresponding expression for the *non*-uniform grid has exactly the same structure with somewhat more complicated coefficients A, B and C. Replacing the partial derivatives with respect to S with their approximations 10.8 and 10.9 for the non-uniform grid in 10.3 yields the following expression for the differential quotient with respect to t evaluated at $t = t_j$

$$
\begin{aligned}
\frac{\partial V(S_i,t_j)}{\partial t} \approx\; & r_j\, W_{i,j} - (r_j - q_j)\, S_i \{ \frac{1}{S_{i+1} - S_{i-1}} [- \frac{S_{i+1} - S_i}{S_i - S_{i-1}} W_{i-1,j} \\
& + \left(\frac{S_{i+1} - S_i}{S_i - S_{i-1}} - \frac{S_i - S_{i-1}}{S_{i+1} - S_i} \right) W_{i,j} + \frac{S_i - S_{i-1}}{S_{i+1} - S_i} W_{i+1,j}] \} \\
& - \frac{1}{2} \sigma_{i,j}^2\, S_i^2 \{ \frac{2}{S_{i+1} - S_{i-1}} [\frac{W_{i-1,j}}{S_i - S_{i-1}} \\
& - \left(\frac{1}{S_i - S_{i-1}} + \frac{1}{S_{i+1} - S_i} \right) W_{i,j} + \frac{W_{i+1,j}}{S_{i+1} - S_i}] \} \\
= \;& \frac{S_i}{(S_{i+1} - S_{i-1})(S_i - S_{i-1})} \left\{ (r_j - q_j)(S_{i+1} - S_i) - \sigma_{i,j}^2\, S_i \right\} W_{i-1,j} \\
& + \{ r_j - \frac{(r_j - q_j)\, S_i}{S_{i+1} - S_{i-1}} \left(\frac{S_{i+1} - S_i}{S_i - S_{i-1}} - \frac{S_i - S_{i-1}}{S_{i+1} - S_i} \right) \\
& + \frac{\sigma_{i,j}^2\, S_i^2}{S_{i+1} - S_{i-1}} \left(\frac{1}{S_i - S_{i-1}} + \frac{1}{S_{i+1} - S_i} \right) \} W_{i,j} \\
& - \frac{S_i}{(S_{i+1} - S_{i-1})(S_{i+1} - S_i)} \left\{ (r_j - q_j)(S_i - S_{i-1}) + \sigma_{i,j}^2\, S_i \right\} W_{i+1,j} \; .
\end{aligned}
$$

Thus, as was the case for the uniform grid

$$
\frac{\partial V(S_i,t_j)}{\partial t} \approx A_{i,j}\, W_{i-1,j} + B_{i,j}\, W_{i,j} + C_{i,j}\, W_{i+1,j} \; , \tag{10.11}
$$

with the slightly more complicated coefficients

$$A_{i,j} = \frac{S_i}{(S_{i+1} - S_{i-1})(S_i - S_{i-1})} \left\{ (r_j - q_j)(S_{i+1} - S_i) - \sigma_{i,j}^2 \, S_i \right\} \tag{10.12}$$

$$B_{i,j} = r_j + \frac{S_i}{(S_{i+1} - S_{i-1})} \{ -(r_j - q_j) \left(\frac{S_{i+1} - S_i}{S_i - S_{i-1}} - \frac{S_i - S_{i-1}}{S_{i+1} - S_i} \right) + \sigma_{i,j}^2 \, S_i \left(\frac{1}{S_i - S_{i-1}} + \frac{1}{S_{i+1} - S_i} \right) \}$$

$$C_{i,j} = -\frac{S_i}{(S_{i+1} - S_{i-1})(S_{i+1} - S_i)} \left\{ (r_j - q_j)(S_i - S_{i-1}) + \sigma_{i,j}^2 \, S_i \right\} .$$

For non-uniform grids as well, the time dependence of the coefficients A, B and C is solely a consequence of the time dependence of r and σ. For time-independent interest rates and volatilities, the coefficients need not have an index j. For uniform grids (at least in the S-direction), in other words, for $S_i = S_0 + i \, \delta S$, the above expression for the coefficients reduces to 10.10.

Substituting this approximation 10.11 for the differential quotient with respect to t in the general equation 10.6 finally gives the generalized form of the finite difference scheme for non-uniform grids:

$$\theta A_{i,j} \, W_{i-1,j} + (\theta B_{i,j} + \frac{1}{t_{j+1} - t_j}) \, W_{i,j} + \theta C_{i,j} \, W_{i+1,j} \tag{10.13}$$

$$\approx \frac{W_{i,j+1}}{(t_{j+1} - t_j)} - (1 - \theta) \left[A_{i,j+1} \, W_{i-1,j+1} + B_{i,j+1} \, W_{i,j+1} + C_{i,j+1} \, W_{i+1,j+1} \right] .$$

As in 10.6, θ can take on arbitrary values between 0 and 1 giving the *implicit* method for $\theta = 1$, the *explicit* method for $\theta = 0$ and the *Crank-Nicolson* method for $\theta = 1/2$. In order to illustrate the importance of this difference equation, note that the values to be calculated on the left-hand side of the equation consist of terms evaluated at time point t_j whereas those on the right-hand side concern only values at time point t_{j+1}, and as such are known, having been calculated in the previous step. Thus, for every (inner) underlying grid point i there is one equation connecting three option values at t_{j+1} with three values at t_j.

The range of the time index j here is always $j = 0, \ldots N-1$ since $j = N$ is already given by the initial condition, i.e., the payoff profile at maturity $T = t_N$. The range of the S-index i (see Eq. 10.1) is at least $i = 1, \ldots, M-1$,

but can take on the values including $i = 0$ and/or $i = M$ (depending on the boundary conditions to be satisfied, see below). The difference equation for $i = 0$ and $i = M$ seems problematic at first glance since values of W are required at points not defined in the grid, for example "$W_{0-1,j}$" or "$W_{M+1,j}$". As will be shown below, these problems can in fact be overcome by consideration of the boundary conditions themselves.

The difference scheme can be written in matrix form (here, for the case where the range of the index i is given by $i = 1, \ldots, M-1$; other cases can be expressed analogously):

$$\underbrace{\begin{pmatrix} a_1 & b_1 & c_1 & 0 & \cdots & \cdots & \ldots & 0 \\ 0 & a_2 & b_2 & c_2 & 0 & & & \vdots \\ \vdots & 0 & a_3 & b_3 & c_3 & 0 & & \vdots \\ \vdots & & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & a_{M-1} & b_{M-1} & c_{M-1} \end{pmatrix}}_{M+1 \text{ columns and } M-1 \text{ rows}} \underbrace{\begin{pmatrix} W_{0,j} \\ W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ W_{M,j} \end{pmatrix}}_{M+1 \text{ rows}} = \underbrace{\begin{pmatrix} D_{1,j} \\ D_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ D_{M-1,j} \end{pmatrix}}_{M-1 \text{ rows}} \tag{10.14}$$

where

$$a_i = \theta A_{i,j}\ ,\ b_i = \theta B_{i,j} + \frac{1}{t_{j+1} - t_j}\ ,\ c_i = \theta C_{i,j}$$

$$D_{i,j} = \frac{W_{i,j+1}}{(t_{j+1} - t_j)} - (1-\theta)\left[A_{i,\ j+1}\, W_{i-1,j+1} + B_{i,j+1}\, W_{i,j+1} + C_{i,j+1}\, W_{i+1,j+1}\right] .$$

The $D_{i,j}$ on the right-hand side depend only on the values at time t_{j+1} which have already been calculated in the previous step and are hence completely determined. For each time step j, we have $(M-1)$ equations for $(M+1)$ unknown $W_{i,j.}$ Other ranges for the index i give the same result: the system of equations is underdetermined; there are two equations fewer than there are unknowns. The two additional equations needed to solve the above system are provided by two boundary conditions.

10.2.1 Initial Conditions

Besides the specification of boundary conditions with respect to the S variable, we also need a boundary condition with respect to the t variable (so called *initial condition*) to be satisfied in order to obtain a unique solution to the above defined difference equation. The initial condition is specified by the *payoff profile* $P(S)$ of the derivative concerned, i.e., by the value $V(S, t_N = T)$ given by

$$V(S, T) = P(S)$$

or in the discrete "grid notation"

$$W_{i,N} = P_i \ .$$

For example, for the payoff profile $P(S)$ for a European call option with strike price K, the initial condition is given by

$$V(S, T) = P(S) \ = \ \max(S - K, 0) \ \ \Longrightarrow \quad W_{i,N} = \max(S_i - K, 0) \ .$$

10.2.2 Dirichlet Boundary Conditions

If either the terms of the option contract (for example, barrier options) or some other information allow us to specify directly the *value* of the option for certain values of S, these values can be used as boundary conditions for the S-grid. Such boundary conditions where the option value itself is given at the boundary are called *Dirichlet boundary conditions*. Let R^U denote these given option values at the *upper boundary* S_M, and R^L denote the given option values at the *lower boundary* S_0, i.e.:

$$V(S_M, t) = R^U(t), \quad V(S_0, t) = R^L(t)$$

or in the discrete "grid notation"

$$W_{M,j} = R^U_j \ , \quad W_{0,j} = R^L_j \ .$$

It is often the case that only approximations for R^U_j and R^L_j are known. If this is the case, the difference scheme provides an approximation which is at best as good as this approximation for the boundary conditions. If, for example,

the Crank-Nicolson scheme is applied but the boundary condition can only be approximated in the first order of S, then the solution procedure as a whole is exact up to first order even though the Crank-Nicolson scheme provides an approximation which is exact up to second order.

Two of the option values to be calculated for each time j are thus specified directly if Dirichlet boundary conditions are given. The dimension of the problem (in the sense of dimension equals number of unknowns) is therefore only $(M-1)$. This is exactly the number of equations in 10.14. Our goal now is to transform Eq. 10.14 into a system consisting of a square matrix, i.e., to reduce the dimension "in each direction" to $(M-1)$. To keep the boundary conditions separate, we write the vector of option values in the form

$$\begin{pmatrix} W_{0,j} \\ W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ W_{M,j} \end{pmatrix} = \begin{pmatrix} 0 \\ W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \\ 0 \end{pmatrix} + \begin{pmatrix} W_{0,j} \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ W_{M,j} \end{pmatrix}.$$

In the first vector, only the (yet to be determined) option values away from the boundary of the grid appear. We call this the *unknown vector*. The second vector contains only the (already specified) values of the option on the boundary. We refer to this vector as the *known vector*. The matrix in Eq. 10.14 acts on both of these vectors. Let us first consider the unknown vector: the only element in the first column of the matrix, namely a_1, acts only on the first row of the unknown vector. This, however, equals zero. We obtain from the matrix multiplication of the unknown vector the same result as if the first column in the matrix and the first row in the vector were removed. The situation is the same for the last column of the matrix and the last row of the unknown vector. Thus we can simply remove the last column of the matrix and the last row of the unknown vector without changing the result of the matrix multiplication.

Matrix multiplication of the full matrix in Eq. 10.14 with the *known* vector is explicitly performed. Combining everything the system of equations in Eq. 10.14 can be equivalently written using an $(M-1)\times(M-1)$ matrix:

$$\underbrace{\begin{pmatrix} b_1 & c_1 & 0 & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & 0 & & \vdots \\ 0 & a_3 & b_3 & c_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & c_{M-2} \\ 0 & \cdots & \cdots & 0 & a_{M-1} & b_{M-1} \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \end{pmatrix}}_{\mathbf{W}_j} + \begin{pmatrix} a_1\, W_{0,j} \\ 0 \\ \vdots \\ \vdots \\ 0 \\ c_{M-1}\, W_{M,j} \end{pmatrix} = \begin{pmatrix} D_{1,j} \\ D_{2,j} \\ \vdots \\ \vdots \\ D_{M-2,j} \\ D_{M-1,j} \end{pmatrix}$$

$$\Leftrightarrow \underbrace{\begin{pmatrix} b_1 & c_1 & 0 & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & 0 & & \vdots \\ 0 & a_3 & b_3 & c_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & c_{M-2} \\ 0 & \cdots & \cdots & 0 & a_{M-1} & b_{M-1} \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \end{pmatrix}}_{\mathbf{W}_j} = \begin{pmatrix} d_{1,j} \\ d_{2,j} \\ \vdots \\ \vdots \\ d_{M-2,j} \\ d_{M-1,j} \end{pmatrix} \tag{10.15}$$

with coeffitions

$$a_i = \theta A_{i,j}\,, \quad b_i = \theta B_{i,j} + \frac{1}{t_{j+1}-t_j}\,, \quad c_i = \theta C_{i,j} \quad \text{for } i = 1,\ldots,M-1$$

$$\begin{aligned} d_{i,j} &= \frac{W_{i,j+1}}{(t_{j+1}-t_j)} \qquad \text{for } i = 2,\ldots,M-2 \\ &\quad - (1-\theta)\left[A_{i,\ j+1}\, W_{i-1,j+1} + B_{i,j+1}\, W_{i,j+1} + C_{i,j+1}\, W_{i+1,j+1}\right] \\ d_{1,j} &= \frac{W_{1,j+1}}{(t_{j+1}-t_j)} - (1-\theta)\left[B_{1,\ j+1}\, W_{1,j+1} + C_{1,j+1}\, W_{2,j+1}\right] \\ &\quad - \left[\theta A_{1,j}\, R_j^L + (1-\theta) A_{1,j+1}\, R_{j+1}^L\right] \\ d_{M-1,j} &= \frac{W_{M-1,j+1}}{(t_{j+1}-t_j)} - (1-\theta)\left[A_{M-1,j+1}\, W_{M-2,j+1} + B_{M-1,j+1}\, W_{M-1,j+1}\right] \\ &\quad - \left[\theta C_{M-1,j}\, R_j^U + (1-\theta) C_{M-1,j+1}\, R_{j+1}^U\right]\,. \end{aligned} \tag{10.16}$$

Along the way, the result of the matrix multiplication with the known vector (i.e., with the boundary condition) has been brought over to the right-hand side. This right-hand side (the $d_{i,j}$) only depends on the values at the time points t_{j+1} (which have already been calculated in the previous step) and on the (given) boundary conditions. The $d_{i,j}$ are thus completely determined. The only unknowns in this system are the $(M-1)$ elements of the vector W_j. These can now be calculated *if* it is possible to invert the matrix $\mathbf{A}$. Because of the special form of the matrix (the only non-zero elements are in the diagonal and the two off-diagonals), a calculation-intensive matrix inversion can be avoided. Instead, we employ a very fast procedure, known as the *L-U decomposition*. To this end, we decompose the matrix $\mathbf{A}$ into the product of a matrix $\mathbf{L}$, which contains non-zero elements only in the diagonal and the "lower" off-diagonal and a matrix $\mathbf{U}$ whose only non-zero elements are found in the diagonal and the "upper" off-diagonal:

$$\mathbf{A} = \underbrace{\begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ l_2 & 1 & 0 & & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & l_{M-2} & 1 & 0 \\ 0 & \cdots\cdots & & 0 & l_{M-1} & 1 \end{pmatrix}}_{\mathbf{L}} \underbrace{\begin{pmatrix} h_1 & u_1 & 0 & \ldots\ldots & 0 \\ 0 & h_2 & u_2 & 0 & \ldots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & \ddots & u_{M-2} \\ 0 & \cdots\cdots\cdots & & 0 & & h_{M-1} \end{pmatrix}}_{\mathbf{U}} .$$

From this *ansatz* the elements of the matrices $\mathbf{L}$ and $\mathbf{U}$ are determined to be

$$\begin{aligned} u_i &= c_i && \text{for } i = 1, \ldots M-2 \\ h_1 &= b_1 \\ l_i &= \tfrac{a_i}{h_{i-1}} && \text{for } i = 2, \ldots M-1 \\ h_i &= b_i - l_i\, u_{i-1} && \text{for } i = 2, \ldots M-1 \,. \end{aligned}$$

Calculating the elements in the order indicated above, we can easily compute all the elements appearing in both matrices. Inverting $\mathbf{L}$ and $\mathbf{U}$ is quite simple. We begin by writing

$$\mathbf{A}\,\mathbf{W}_j = \mathbf{L}\underbrace{\mathbf{U}\,\mathbf{W}_j}_{\mathbf{x}} =: \mathbf{L}\mathbf{x}$$

and solve the system for our newly defined vector **x:**

$$\begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ l_2 & 1 & 0 & & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & l_{M-2} & 1 & 0 \\ 0 & \cdots & \cdots & 0 & l_{M-1} & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_{M-1} \end{pmatrix} = \begin{pmatrix} d_{1,j} \\ d_{2,j} \\ d_{3,j} \\ \vdots \\ \vdots \\ d_{M-2,j} \\ d_{M-1,j} \end{pmatrix} .$$

To solve this system for **x**, we start with the first row of the matrix which contains only one non-zero element and proceed from top to bottom to obtain

$$\begin{aligned} x_1 &= d_{1,j} \\ x_i &= d_{i,j} - l_i\, x_{i-1} \quad for i = 2 \ldots M-1 \; . \end{aligned}$$

Now, with vector **x** is known, the option values $W_{i,j}$ can simply be calculated from the above definition of **x** by solving the equation $\mathbf{U}\ W_j = \mathbf{x}$:

$$\begin{pmatrix} h_1 & u_1 & 0 & \cdots & \cdots & 0 \\ 0 & h_2 & u_2 & 0 & \ldots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & \ddots & u_{M-2} \\ 0 & \cdots & \cdots & \cdots & 0 & h_{M-1} \end{pmatrix} \begin{pmatrix} W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_{M-1} \end{pmatrix} .$$

To "invert" the matrix **U**, we begin this time with the last row of the matrix and work our way up to obtain

$$\begin{aligned} W_{M-1,j} &= \frac{x_{M-1}}{h_{M-1}} \\ W_{i,j} &= \frac{x_i - u_i W_{i+1,j}}{h_i} \quad \text{for} \quad i = 1 \ldots M-2 \end{aligned} \tag{10.17}$$

Thus, we have found an explicit solution for the general finite difference scheme 10.13, expressing the option values at time t_j in terms of the values at a later time point t_{j+1} under consideration of the given values at the boundaries.

We will now consider the plain vanilla put and call as well as the barrier option as examples for the application of this scheme with Dirichlet boundary conditions. While, in principle, we must find the values of the plain vanilla options on the entire set from $S = 0$ to $S = \infty$, the value of the barrier option is specified directly in the option contract on one side of the barrier (and on the barrier itself). The area to be covered by its grid is thus smaller than that for plain vanilla options.

Let us, however, begin by considering plain vanilla options. For these options, the boundary conditions must be specified at "$S = 0$" and "$S = \infty$". We choose a lower bound S_0 small enough, such that the value of the *call* for such values of the underlying price is negligible (the option is well out of the money). Furthermore, we choose an upper bound S_M large enough, such that the value of a *put* is negligible for such values of the underlying price (the option is well out of the money). The put-call parity is then used to establish the other boundary values for each of the two options. Recall that for European options on an underlying paying a dividend yield q the put-call parity, Eq. 6.8, is given by:

$$\text{Price(Call)} - \text{Price(Put)} = \text{Price(Forward)} = Se^{-q(T-t)} - Ke^{-r(T-t)} .$$

The right-hand side is the value of a forward contract[3] with strike price K. Since the put at the upper boundary is worthless, the call option has the same value as the forward contract (the option is so far in the money that it will be exercised with certainty). Conversely, the call at the lower boundary is worthless, therefore the put has the same value as a short forward contract (the option is so far in the money that it will be exercised with certainty). Summarizing, we have the following Dirichlet boundary conditions for the plain vanilla options:

	$S_0 \approx 0$	$S_M \approx \infty$
call	$R_j^L = 0$	$R_j^U = S_M e^{-q(T-t_j)} - Ke^{-r(T-t_j)}$
put	$R_j^L = -S_0 e^{-q(T-t_j)} + Ke^{-r(T-t_j)}$	$R_j^U = 0$

[3]For the sake of simplicity, the price of the forward contract is given for the case of a flat interest rate term structure, a flat dividend yield curve and no discrete dividend payments. However, this relation also holds for interest rates and dividend yields which are time-dependent.

We now take an up-and-out barrier call option as an example of a somewhat exotic and path dependent option and specify its boundary condition. The initial condition and the boundary condition for $S = 0$ is exactly the same as that of a plain vanilla call but as soon as S attains the upper barrier level H, the option becomes worthless. Often, the option's holder still receives a payment if the barrier is reached, the so-called *rebate*. Let us assume that the rebate R is due at precisely the knock-out time point (i.e., the instant when S touches or breaches the barrier). Then, the option's value is given by the value of the rebate, namely R. It is thus convenient to select the upper boundary of the grid as $S_M = H$. The boundary condition for such a barrier option is then simply $W_{M,j} = R$.

10.2.3 Neumann Boundary Condition

It often occurs that the first derivative of the option price S is specified at the boundary of the grid rather than the option price itself,

$$\left.\frac{\partial V(S,t)}{\partial S}\right|_{S=S_M} = R^U(t), \quad \left.\frac{\partial V(S,t)}{\partial S}\right|_{S=S_0} = R^L(t) \ .$$

Boundary conditions of this type are called *Neumann boundary conditions*.[4] Since these boundary conditions do not directly specify the *values* of the solution (the option values) on the boundary, all $(M + 1)$ values must be calculated in each time slice. The dimension (the number of values to be calculated) of the problem is thus $(M + 1)$. In order to obtain the same number of equations, the index i in Eq. 10.13 must range from 0 to M. As a result, two "additional unknowns", namely "$W_{-1,j}$" and "$W_{M+1,j}$" appear in the system of equations. The two Neumann boundary conditions will be used to eliminate these additional unknowns. In comparison to the Dirichlet conditions, we will then have system of equations consisting of two more equations but exactly as many equations as unknowns.

Since we use three point finite differences in S-direction (see for example Eq. 10.8) the grid must be extended at both boundaries by one virtual grid point in S-direction for each index j. In other words, we add two additional grid points S_{-1} and S_{M+1} at each time point t_j. This means that the index i in

[4] As already pointed out in the section on Dirichlet boundary conditions, the solution is at best as exact as the given boundary conditions. This is important in all cases where there are only approximations to the boundary conditions $R^U(t)$ and $R^L(t)$ available.

the difference equation 10.13 takes on the values $i = 0, \ldots, M$. In the finite difference approximation 10.8 of the first derivative with respect to S, we can substitute the respective Neumann conditions for the cases $i = 0$ and $i = M$ on the left-hand side of the equation and then rearrange the terms so that the option values outside the grid are expressed in terms of those defined within the grid and the boundary conditions. This yields

$$W_{M+1,j} \approx \left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2 W_{M-1,j} + \left[1 - \left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2\right] W_{M,j} + \frac{S_{M+1} - S_M}{S_M - S_{M-1}}(S_{M+1} - S_{M-1}) R_j^U$$

at the upper boundary, i.e., for $i = M$. Likewise for $i = 0$ on the lower boundary we have

$$W_{-1,j} \approx \left[1 - \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2\right] W_{0,j} + \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2 W_{1,j} - \frac{S_0 - S_{-1}}{S_1 - S_0}(S_1 - S_{-1}) R_j^L \,.$$

Substituting these expressions into Eq. 10.11, we obtain the approximation for the differential quotient with respect to *time* at the boundary of the S-grid. For the upper boundary, this gives:

$$\begin{aligned}
\frac{\partial V(S_M, t_j)}{\partial t} &\approx A_{M,j}\, W_{M-1,\,j} + B_{M,j}\, W_{M,j} \\
&\quad + C_{M,j} \{\left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2 W_{M-1,j} + \left[1 - \left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2\right] W_{M,j} \\
&\quad + \frac{S_{M+1} - S_M}{S_M - S_{M-1}}(S_{M+1} - S_{M-1}) R_j^U\} \\
&= A^*_{M,j}\, W_{M-1,j} + B^*_{M,j}\, W_{M,j} \\
&\quad + \frac{S_{M+1} - S_M}{S_M - S_{M-1}}(S_{M+1} - S_{M-1}) R_j^U\, C_{M,j}
\end{aligned}$$

where

$$A^*_{M,j} = A_{M,j} + \left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2 C_{M,j}$$

$$B^*_{M,j} = B_{M,j} + \left[1 - \left(\frac{S_{M+1} - S_M}{S_M - S_{M-1}}\right)^2\right] C_{M,j} \ .$$

Likewise for the lower boundary:

$$\begin{aligned}\frac{\partial V(S_0, t_j)}{\partial t} &\approx A_{0,j}\{\left[1 - \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2\right] W_{0,\,j} + \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2 W_{1,j} \\ &\quad - \frac{S_0 - S_{-1}}{S_1 - S_0}(S_1 - S_{-1}) R^L_j\} + B_{0,j}\, W_{0,\,j} + C_{0,j}\, W_{1,j} \\ &= B^*_{0,j}\, W_{0,j} + C^*_{0,j}\, W_{1,j} - \frac{S_0 - S_{-1}}{S_1 - S_0}(S_1 - S_{-1}) R^L_j A_{0,j}\end{aligned}$$

where

$$B^*_{0,j} = B_{0,j} + \left[1 - \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2\right] A_{0,j}$$

$$C^*_{0,j} = C_{0,j} + \left(\frac{S_0 - S_{-1}}{S_1 - S_0}\right)^2 A_{0,j} \ .$$

All preparations have now been made for specifying the difference scheme 10.13 at the boundaries. Replacing the differential quotient with respect to *time* in Eq. 10.6 with the above approximation gives

$$\begin{aligned}&\theta A^*_{M,j}\, W_{M-1,j} + (\theta B^*_{M,j} + \frac{1}{t_{j+1} - t_j})\, W_{M,j} \\ &\approx \frac{W_{M,j+1}}{(t_{j+1} - t_j)} - (1 - \theta)\left[A^*_{M,j+1}\, W_{M-1,j+1} + B^*_{M,j+1}\, W_{M,j+1}\right] \\ &- \frac{S_{M+1} - S_M}{S_M - S_{M-1}}(S_{M+1} - S_{M-1})\left[\theta R^U_j\, C_{M,j} + (1 - \theta) R^U_{j+1}\, C_{M,j+1}\right]\end{aligned}$$

for the upper boundary. Likewise, the difference scheme for the lower boundary is obtained as

$$
\begin{aligned}
&(\theta B^*_{0,j} + \frac{1}{t_{j+1} - t_j})\, W_{0,j} + \theta C^*_{0,\,j}\, W_{1,j} \\
&\approx \frac{W_{0,j+1}}{(t_{j+1} - t_j)} - (1-\theta)\left[B^*_{0,j+1}\, W_{0,j+1} + C^*_{0,j+1}\, W_{1,j+1}\right] \\
&+ \frac{S_0 - S_{-1}}{S_1 - S_0}(S_1 - S_{-1})\left[\theta R^L_j A_{0,j} + (1-\theta) R^L_{j+1} A_{0,j+1}\right] .
\end{aligned}
$$

Combining the above results, the finite difference scheme 10.13 for Neumann boundary conditions can be written in the following matrix form:

$$
\underbrace{\begin{pmatrix}
b_0 & c_0 & 0 & 0 & 0 & 0 & \dots & 0 \\
a_1 & b_1 & c_1 & 0 & 0 & 0 & \dots & 0 \\
0 & a_2 & b_2 & c_2 & 0 & 0 & \dots & 0 \\
0 & 0 & a_3 & b_3 & c_3 & 0 & \dots & 0 \\
0 & 0 & 0 & \ddots & \ddots & \ddots & & 0 \\
\vdots & & & \ddots & \ddots & \ddots & \ddots & 0 \\
\vdots & & & & 0 & a_{M-1} & b_{M-1} & c_{M-1} \\
0 & \dots & \dots & \dots & \dots & 0 & a_M & b_M
\end{pmatrix}}_{\mathbf{A}}
\underbrace{\begin{pmatrix} W_{0,j} \\ W_{1,j} \\ W_{2,j} \\ \\ \\ \\ W_{M,j} \end{pmatrix}}_{\mathbf{W}_j}
=
\underbrace{\begin{pmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ \\ \\ \\ d_{M,j} \end{pmatrix}}_{\mathbf{d}_j}
\tag{10.18}
$$

where

$$
a_i = \theta A_{i,j}\ ,\quad b_i = \theta B_{i,j} + \frac{1}{t_{j+1} - t_j}\ ,\quad c_i = \theta C_{i,j} \ \text{ for } \ i = 1, \dots, M-1
$$

$$
\begin{aligned}
d_{i,j} &= \frac{W_{i,j+1}}{(t_{j+1} - t_j)} \qquad\qquad \text{for } \ i = 1, \dots, M-1 \\
&\quad - (1-\theta)\left[A_{i,\,j+1}\, W_{i-1,j+1} + B_{i,j+1}\, W_{i,j+1} + C_{i,j+1}\, W_{i+1,j+1}\right]
\end{aligned}
$$

$$
a_M = \theta A^*_{M,j}\ ,\quad b_M = \theta B^*_{M,j} + \frac{1}{t_{j+1} - t_j},
$$

$$
\begin{aligned}
d_{M,j} &= \frac{W_{M,j+1}}{(t_{j+1} - t_j)} - (1-\theta)\left[A^*_{M,\,j+1}\, W_{M-1,j+1} + B^*_{M,j+1}\, W_{M,j+1}\right] \\
&\quad - \frac{S_{M+1} - S_M}{S_M - S_{M-1}}(S_{M+1} - S_{M-1})\left[\theta R^U_j\, C_{M,j} + (1-\theta) R^U_{j+1}\, C_{M,j+1}\right]
\end{aligned}
$$

$$
\begin{aligned}
b_0 &= \theta B^*_{0,j} + \frac{1}{t_{j+1} - t_j}\,, \quad c_0 = \theta C^*_{0,j} \\
d_{0,j} &= \frac{W_{0,j+1}}{(t_{j+1} - t_j)} - (1-\theta)\left[B^*_{0,\ j+1}\, W_{0,j+1} + C^*_{0,j+1}\, W_{1,j+1}\right] \\
&\quad + \frac{S_0 - S_{-1}}{S_1 - S_0}(S_1 - S_{-1})\left[\theta R^L_j A_{0,j} + (1-\theta) R^L_{j+1} A_{0,j+1}\right] .
\end{aligned}
\tag{10.19}
$$

This system of equations must be solved for each time step taking the boundary conditions into consideration. The right-hand side of Eq. 10.18 consists only of values evaluated at time point t_{j+1} (which have already been computed in the previous step) and of the (given) boundary conditions. The $d_{i,j}$ are thus completely determined. The system of equations has the same structure as in the corresponding system 10.15 for the Dirichlet problem except that it is two dimensions larger. Here, $\mathbf{A}$ is a $(M+1) \times (M+1)$ matrix. This is explained by the fact that the grid has been extended at the upper and lower boundary in the S-direction. The simple structure of the matrix $\mathbf{A}$ again allows to perform a L-U decomposition. As before, we begin by setting $\mathbf{A}\ W_j = \mathbf{L}\,\mathbf{U}\ W_j =: \mathbf{L}\mathbf{x}$ and, exactly as was the case for Dirichlet boundary conditions, we obtain

$$
\begin{aligned}
u_i &= c_i && \text{for } i = 0, \ldots M-1 \\
h_0 &= b_0 \\
l_i &= \frac{a_i}{h_{i-1}} && \text{for } i = 1, \ldots M \\
h_i &= b_i - l_i\, u_{i-1} && \text{for } i = 1, \ldots M \\
x_0 &= d_{0,j} \\
x_i &= d_{i,j} - l_i\, x_{i-1} && \text{for } i = 1 \ldots M \\
W_{M,j} &= \frac{x_M}{h_M} \\
W_{i,j} &= \frac{x_i - u_i W_{i+1,j}}{h_i} && \text{for } i = 0 \ldots M-1\ .
\end{aligned}
\tag{10.20}
$$

The only difference is that the range of the index i has been extended and that a_i, b_i, c_i, d_i are now given by Eq. 10.19.

As an example of the Neumann problem, we consider the plain vanilla put and call. Neumann boundary conditions are usually easier to determine than Dirichlet conditions. For S_0 the call is well out of the money and

the dependence of the price on S can be assumed to be negligible. The corresponding boundary condition is thus $\frac{\partial V(S,t)}{\partial S}\Big|_{S=S_0} = 0$. For extremely large S, the call is way in the money and thus its behavior is approximated by the payoff profile. The derivative in the S direction is thus 1 with the corresponding boundary condition given by $\frac{\partial V(S,t)}{\partial S}\Big|_{S=S_0} = 1$. The behavior of the put is the exact opposite. Summarizing, the Neumann boundary conditions for the plain vanilla put and call are given by:

	$S_0 \approx 0$	$S_M \approx \infty$	$t_N = T$
call	$R_j^L = 0$	$R_j^U = 1$	$P_i = \max(S_i - K, 0)$
put	$R_j^L = -1$	$R_j^U = 0$	$P_i = \max(K - S_i, 0)$

For the sake of completeness, we have also included the initial conditions in the table.

The type of boundary condition selected for the analysis depends on the type of option being priced. For plain vanilla options, Neumann boundary conditions are attractive because they can be easily calculated. The pricing of a forward contract at each time point (which must be done to establish the Dirichlet boundary conditions), on the other hand, can be tedious, particularly when the term structure is not flat and discrete dividends must be taken into consideration.

For Knock-out barrier options, however, the exact value of the option at the barrier is known (0 or the rebate), while the change of its value at the boundary is unknown. In this case, it makes sense to use the Dirichlet boundary conditions. In addition, a combination of Dirichlet and Neumann boundary conditions can be applied to one and the same problem as needed.

10.2.4 Generalized Neumann Boundary Conditions

It is possible to generalize the Neumann boundary condition by a simple method that does not need any explicit information on boundary conditions with respect to S. Valuation of derivatives most often deals with diffusion problems, with the Black Scholes equation being its most prominent example. Diffusion problems share the property that the pay off profile is being smoothed or "smeared out" the more the farther away the expiry is in the future. At the same time, the pay off profile is a simple linear function of S for

very small or very large S. Consequently, at the boundaries, the fair value can be approximated as a linear function of S.[5]

A linear function has the property that the second derivative is zero. Therefore, instead of specifying the value of the first derivative at the boundary, which is a standard Neumann condition, we may specify the value of the second derivative, which is (almost) independent of the option type (i.e. it is equal to zero) as a *generalized* Neumann condition.

We assume a grid in dimension S with $M+1$ nodes $S_0, S_1, S_2, \ldots, S_M$, with S_0 and S_M being virtual grid nodes. They will be derived from the constraint of zero second order derivatives at the boundaries and thus eliminated.

First consider the lower boundary $i = 1$. We calculate the unknown value $W_{0,j}$ by setting Eq. 10.9 to zero for the second derivative with respect to S at node $i = 1$:

$$\frac{2}{S_2 - S_0}\left[\frac{W_{0,j}}{S_1 - S_0} - \left(\frac{1}{S_1 - S_0} + \frac{1}{S_2 - S_1}\right) W_{1,j} + \frac{W_{2,j}}{S_2 - S_1}\right] = 0\ . \tag{10.21}$$

The virtual grid node S_0 may be chosen freely. For simplicity, we set $S_1 - S_0 = S_2 - S_1$, i.e. $S_0 = 2S_1 - S_2$. Then, all terms in Eq. 10.21 have the same denominator. If follows:

$$\begin{aligned} W_{0,j} - 2W_{1,j} + W_{2,j} &= 0 \\ \Leftrightarrow \quad W_{0,j} &= 2W_{1,j} - W_{2,j}\ . \end{aligned}$$

An analogous expression can be derived for node M:

$$\Leftrightarrow \quad W_{M,j} = 2W_{M-1,j} - W_{M-2,j}\ .$$

Now, the problem is reduced to a system of Dirichlet type (Eq. 10.14) with known $W_{0,j}$ and $W_{M,j}$. Alternatively, it is also possible to set a_1 in 10.15 to zero and replace b_1 by $\tilde{b}_1 = b_1 + 2a_1$ and c_1 by $\tilde{c}_1 = c_1 - a_1$. With this, $W_{0,j}$

[5] One of the rare exceptions to this rule is a power option without cap, with a pay off profile proportional to S^2. Such options are of minor relevance in praxis.

has been eliminated. Analogous, $W_{M,j}$ can be eliminated as well. This is the same as replacing Eq. 10.15 by

$$\begin{pmatrix} b_1+2a_2 & c_1-a_1 & 0 & \cdots & \cdots & 0 \\ a_2 & b_2 & c_2 & 0 & & \cdots \\ 0 & a_3 & b_3 & c_3 & \ddots & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & c_{M-2} \\ 0 & \cdots & \cdots & 0 & a_{M-1}-c_{M-1} & b_{M-1}+2c_{M-1} \end{pmatrix} \begin{pmatrix} W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \end{pmatrix} = \begin{pmatrix} D_{1,j} \\ D_{2,j} \\ \vdots \\ \vdots \\ D_{M-2,j} \\ D_{M-1,j} \end{pmatrix}$$

with the same coefficients as in Eq. 10.16.

10.2.5 Free Boundary Conditions for American Options

Thus far our discussion has been restricted to European options. Finite difference methods can also be extended to American options. The right to early exercise must, of course, be taken into account in pricing the option. The Black-Scholes equation holds only on a restricted set and must be replaced by the two (in)equalities given by Eqs. 7.13 and 7.14. Equivalently expressed, the Black-Scholes equation 7.13 holds on a parameter set with a free boundary given by $S^*(t)$. This line $S^*(t)$ separates the parameter set where the Black-Scholes *equation* holds from the parameter set where the value of the option is simply given by the payoff profile 7.14. Directly on this curve $S^*(t)$ we have

$$V(S^*(t), t) = P(S^*(t)) \ .$$

This is a Dirichlet boundary condition. The boundary condition is called *free* because $S^*(t)$ is not known a priori.

A simple solution for this problem can be found using the *explicit* finite difference procedure. As for the European option, we start by determining a solution vector $W_{i,j}$, $i = 0, 1, \ldots, M$ for a time step j. An intermediate calculation is performed to determine the solution vector $\widetilde{W}_{i,j}$ for the American option. In this calculation, the European price is determined and compared to the intrinsic value $P(S)$. The value of the American option is then defined as the larger of the two, i.e.,

$$\widetilde{W}_{i,j} = \max\left[W_{i,j}, P(S_i)\right] \ .$$

The procedure continues with the next time step using this solution vector.

If an implicit (or mixed) finite difference procedure is utilized, this will *not* be exact since the vector $W_{i,j}$ will already have been used implicitly in its calculation. Therefore the procedure can in principle not be split into the above two steps. It can be shown, however, that the error in doing so remains small if the grid in t is fine enough. In particular, if the distance between two time points is one day or less, the error is often negligible. An additional trick can be used for call options. It is known that the exercise of a call option can only be optimal (if at all) immediately before the payment of a discrete dividend. We can thus introduce extra time points on and immediately preceding the due dates of the dividend payments with a time difference of, for example, half a day, and thus minimize the error.

In any case, in addition to the free boundary conditions, values for the upper and lower boundaries of the S-grid must also be specified. This is already clear from the explicit procedure described above. We can obviously only compare the solution vector for a European option with the payoff profile if this vector has already been determined. To determine this solution vector for the European option, we need two boundary conditions. They can be either Neumann, Dirichlet, or more general boundary conditions (such as the second derivative equals zero). For an American option all boundary conditions involving derivatives are generally the same as for the corresponding European option. The maximum of the payoff profile and the boundary condition of the corresponding European option is a good candidate for the Dirichlet boundary condition. Thus, for American plain vanilla put and call options:

	Call
$S_0 \approx 0$	$R_j^L = 0$
$S_M \approx \infty$	$R_j^U = \max\left[S_M e^{-q(T-t_j)} - K e^{-r(T-t_j)}, S_M - K\right]$
$t_N = T$	$P_i = \max(S_i - K, 0)$
	Put
$S_0 \approx 0$	$R_j^L = \max\left[-S_0 e^{-q(T-t_j)} + K e^{-r(T-t_j)}, -S_0 + K\right]$
$S_M \approx \infty$	$R_j^U = 0$
$t_N = T$	$P_i = \max(K - S_i, 0)$

Note that for American options this is not enough. We also need to specify the conditions on the free boundary $S^*(t)$. And before we can do this, we need to determine $S^*(t)$ in the first place.

The Lamberton and Lapeyre Procedure

In many cases, the free boundary lies on only one side of the grid. For this case, a suitable procedure for evaluating the American put options has been suggested by *Lamberton* and *Lapeyre* [129]. This procedure does not necessitate any appreciable additional computational effort. It simply varies the order of operations needed to solve the system of equations for one time step. We will now present this procedure.

The Dirichlet boundary condition for the plain vanilla option as given above will be used as the specified boundary condition (other types of boundary conditions could also have been used). The difference scheme then has the form given in Eq. 10.15 with coefficients given by Eq. 10.16. Before introducing the procedure, we consider the following limiting cases: the exercise region for a put option must lie in regions where the value of S is small since for large S (for $S > K$) the intrinsic value of a put is zero; the option only has a time value. For calls (if at all), the exercise region must lie in regions where S is large since for small S (for $S < K$), the intrinsic value of the call is zero.

We now consider the iteration scheme 10.17. In this scheme, we start by determining the value of $W_{M-1,j}$. This is an option value for large S. From the above consideration we know that an optimal early exercise of an American *call* option cannot happen in the small S region, but *could* happen in the large S region. Therefore we have to allow for this possibility at the grid point (S_{M-1}, t_j). Should it indeed be optimal to exercise in this time step then the value $W_{M-1,j}$ must be replaced by the intrinsic value. This fact can be incorporated into our scheme using the expression

$$\widetilde{W}_{M-1,j} = \max\left[\frac{x_{M-1}}{h_{M-1}}, S_{M-1} - K\right]$$

for our calculation of $\widetilde{W}_{M-1,j}$ instead of 10.17. We thus obtain the correct boundary value $\widetilde{W}_{M-1,j}$ for the American call option. In the next step we use

$$\widetilde{W}_{M-2,j} = \max\left[\frac{x_{M-2} - u_{M-2}\widetilde{W}_{M-1,j}}{h_{M-2}}, S_{M-2} - K\right]$$

instead of the corresponding expression 10.17 for the European option. Likewise, this leads to the correct result since the value $\widetilde{W}_{M-1,j}$ for an American option was calculated correctly before. This means, however, that an American

call option is correctly computed even with implicit or mixed methods if we simply substitute

$$\widetilde{W}_{i,j} = \max\left[\frac{x_i - u_i \widetilde{W}_{i+1,j}}{h_i}, S_i - K\right]$$

for $i = M-2, M-3, \ldots, 2, 1$ at every time step. The procedure just described started with large S (Index $i = M-1$) and computes step-wise to smaller values of S. This makes it only suitable for *call* options.

The procedure must be modified for *puts*. Instead of an **LU** decomposition we now carry out a **UL** decomposition, for which the upper and lower triangular matrices appear in reverse order. The decomposition of the coefficient matrix **A** is then given by:

$$\mathbf{A} = \underbrace{\begin{pmatrix} h_1 & u_1 & 0 & \cdots & \cdots & 0 \\ 0 & h_2 & u_2 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & \ddots & u_{M-2} \\ 0 & \cdots & \cdots & 0 & h_{M-1} & \end{pmatrix}}_{\mathbf{U}} \underbrace{\begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ l_2 & 1 & 0 & & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & l_{M-2} & 1 & 0 \\ 0 & \cdots & \cdots & 0 & l_{M-1} & 1 \end{pmatrix}}_{\mathbf{L}}$$

The matrix elements differ only slightly for those given by the **LU** decomposition:

$$\begin{aligned} u_i &= c_i && \text{for } i = 1, \ldots M-2 \\ h_{M-1} &= b_{M-1} && \\ l_i &= \tfrac{a_i}{h_i} && \text{for } i = 2, \ldots M-1 \\ h_i &= b_i - l_{i+1}\, u_i && \text{for } i = 1, \ldots M-2 \end{aligned}$$

The decisive difference is that h_i is iteratively determined from large values down to small ones (instead of small to large). Proceeding in this way, the matrix elements of **U** and **L** can be easily calculated. Analogously to the case of the **LU** decomposition we set

$$\mathbf{A}\,\mathbf{W}_j = \mathbf{U}\underbrace{\mathbf{L}\,\mathbf{W}_j}_{\mathbf{x}} =: \mathbf{U}\mathbf{x}$$

and first solve the system of equations for the new vector **x:**

$$\begin{pmatrix} h_1 & u_1 & 0 & \cdots & \cdots & 0 \\ 0 & h_2 & u_2 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & \ddots & u_{M-2} \\ 0 & \cdots & \cdots & \cdots & 0 & h_{M-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_{M-1} \end{pmatrix} = \begin{pmatrix} d_{1,j} \\ d_{2,j} \\ d_{3,j} \\ \vdots \\ \vdots \\ d_{M-2,j} \\ d_{M-1,j} \end{pmatrix} .$$

To compute **x** we start with the row in the matrix which contains only one non-zero element, i.e., the last row, and proceed from bottom to top to obtain

$$\begin{aligned} x_{M-1} &= \frac{d_{M-1,j}}{h_{M-1}} \\ x_i &= \frac{d_{i,j} - u_i\, x_{i+1}}{h_i} \quad \text{for} \quad i = 1 \ldots M-2 \, . \end{aligned}$$

Now that **x** is known, the desired option values $W_{i,j}$ can be calculated from the definition of **x**, solving the equation $\mathbf{L}\ W_j = \mathbf{x}$:

$$\begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ l_2 & 1 & 0 & & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & l_{M-2} & 1 & 0 \\ 0 & \cdots & \cdots & 0 & l_{M-1} & 1 \end{pmatrix} \begin{pmatrix} W_{1,j} \\ W_{2,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,j} \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_{M-1} \end{pmatrix} .$$

To "invert" the matrix **L**, we start with the row containing only one non-zero element, working from top to bottom to obtain

$$\begin{aligned} W_{1,j} &= x_1 \\ W_{i,j} &= x_i - l_i W_{i-1,j} \quad \text{for} \quad i = 2 \ldots M-1 \, . \end{aligned}$$

Now, the iteration begins with the small index values, i.e., with small S-values. This is what we need since from the limit considerations above we know that

optimal early exercise of an American put option can only happen in the small S region. Therefore we have to start in this region to "capture" the early exercise possibilities correctly. This is completely analogous to the case of an American call presented above, where we also started in the region where early exercise could be possible (it was the large S region in that case).

Consequently, we now obtain correct values of American put options for implicit and mixed methods by setting:

$$\widetilde{W}_{1,j} = \max\left[x_1,\, S_1 - K\right]$$
$$\widetilde{W}_{i,j} = \max\left[x_i - l_i \widetilde{W}_{i-1,j}\,,\; S_i - K\right] \quad \text{for} \quad i = 2 \ldots M-1\,.$$

10.3 Convergence Criteria

As for all numerical methods, the essential question to be answered before implementing finite differences is whether the procedure is stable, i.e., whether the numerical solution in fact converges towards the actual solution. To motivate this complex subject, we first consider as an example the special case $\theta = 0$, i.e., the *explicit* method. As can be seen from Eqs. 10.16 and 10.19, the off-diagonal terms are zero regardless of the type of boundary conditions used. This implies that the system of equations is completely uncoupled. For example, for Eq. 10.16, the system of equations reduces to

$$\begin{aligned}
W_{0,j} &= \left(1 - (t_{j+1} - t_j)\left(B_{1,j+1} + 2A_{1,j+1}\right)\right) W_{1,\,j+1} \\
&\quad - (t_{j+1} - t_j)\left(C_{1,j+1} - A_{1,j+1}\right) W_{2,j+1} \\
W_{i,j} &= \left(1 - (t_{j+1} - t_j) B_{i,j+1}\right) W_{i,\,j+1} \qquad \text{for } i = 1 \ldots M-1 \\
&\quad - (t_{j+1} - t_j) A_{i,j+1}\, W_{i-1,j+1} - (t_{j+1} - t_j) C_{i,j+1}\, W_{i+1,j+1} \\
W_{M-1,j} &= \left(1 - (t_{j+1} - t_j)\left(B_{M-1,j+1} + 2C_{M-1,j+1}\right)\right) W_{M-1,\,j+1} \\
&\quad - (t_{j+1} - t_j)\left(A_{M-1,j+1} - C_{M,j+1}\right) W_{M-2,j+1}\,.
\end{aligned} \tag{10.22}$$

This means that we obtain the unknown values at time point t_j directly (without carrying out any matrix inversion!) in terms of the values already calculated for time point t_{j+1}. This result can also be obtained much simpler and more directly by starting with Eq. 10.4 for the explicit method rather than from the more general expression 10.6.

Table 10.1 Properties of the three most commonly used finite difference methods

Scheme	θ	Numerical effort	Convergence	Stability
Implicit	1.0	Large	Slow	Unlimited
Explicit	0.0	Minor	Slow	Limited
Crank-Nicolson	0.5	Large	Fast	Unlimited

At this point the reader could ask why we have taken the trouble of introducing the generalized expression involving θ when the explicit method is so simple. For this we have to understand how the parameter θ influences the properties of the finite difference scheme. An important criterion is stability. Table 10.1 contains an overview of the three most commonly used values of θ.

A difference scheme is called *stable* if small deviations from the correct solution do *not* grow arbitrarily fast (no faster than exponentially) with time.[6] If a difference scheme is not stable, errors in the approximation arising in an iteration step are amplified by calculating backwards in the time-grid from one iteration step to the next. Obviously, a difference scheme is only useful if it is stable, since deviations from the correct solution would otherwise grow faster than exponentially when calculating backwards through the grid, finally yielding a result with an arbitrarily large error. It can be shown that for $\theta \geq 0.5$, the finite difference scheme is always stable, independent of the choice of grid.

A difference scheme *converges* towards the correct solution if it is both *stable* and *consistent*. *Consistency* is satisfied if the difference scheme applied to a smooth function at a fixed time point gives an arbitrarily good approximation of the solution to the differential equation by making the grid fine enough. Consistency thus means "convergence for one time step". If the system is consistent and stable, i.e., the growth of the error over one iteration step is bounded, the scheme converges *entirely*, i.e., over all time steps.

There exists a criterion for the stability of an *explicit* difference scheme ($\theta = 0$) for the well-known heat equation 7.22 which, as we have seen, is closely related to the Black-Scholes equation. This criterion is (expressed in the variables from Eq. 7.22):

$$\frac{\delta\tau}{\delta x^2} \leq \frac{1}{2}\,.$$

[6] Independent of the number of time steps.

For the Black-Scholes equation, the stability condition is considerably more complicated [181]:

$$\frac{1}{2}\left[(r(t)-q(t))S\frac{\delta t}{\delta S}\right]^2 \leq \frac{1}{2}\sigma(S,t)^2 S^2 \frac{\delta t}{(\delta S)^2} \leq \frac{1}{2}\,. \tag{10.23}$$

We are, in fact, now dealing with two inequalities. The inequality appearing on the left in the above expression is generally satisfied for the usual values for r, q and σ. The second inequality however, poses a strong restriction in the application of the explicit method since it requires the number of time steps to increase quadratically with the number of the S-steps (i.e., as δS becomes smaller). A consequence of the S-dependence of this criterion is that the difference scheme can be stable on one part of the grid and unstable on the other. Such local instabilities often go unnoticed, but lead to incorrect results; for example, negative prices could arise. Such an example can be found in Hull [103]. The criterion 10.23 for stability can be generalized further: the generalized difference scheme for non-uniform grids remains stable for $\theta < 0.5$ when the following holds for the second inequality:

$$\frac{(t_{j+1}-t_j)}{(S_{i+1}-S_i)^2}\sigma_{i,j}^2 S_i^2 \leq \frac{1}{2(1-2\theta)}\,, \quad \theta < \frac{1}{2}\,.$$

The Crank-Nicolson method $\theta = 0.5$ has an advantage over the implicit and explicit methods in that it converges considerably faster. Because of the averaging of the finite differences at times t_j and t_{j+1}, this method is exact up to second order although the time derivative corresponds to only a first order approximation. Achieving the same degree of accuracy using the implicit method would require considerably more time steps. The Crank-Nicolson method loses some of its efficient convergence when the initial and/or boundary conditions become less smooth. In many applications, for instance, the first derivative of the initial condition is not continuous. This is already the case for the plain vanilla option (at the strike). However, the discontinuity at that point leads to a negligible inaccuracy in the option price. A significant error, however, can be found for the value of the delta, while the values for gamma at the point of the discontinuity can be so extreme that the error may exceed 100%. As a result, it may be the case that the gamma for the plain vanilla option cannot be computed with sufficient accuracy when the spot price equals the strike price (the first derivative of the payoff is discontinuous at this point). In such situations the implicit method is often the only suitable method for calculating gamma with sufficient accuracy.

Another possibility would be to use a three-time-level approximation of the time derivative which also has the effect of smoothing the oscillations. A detailed description of this and other procedures can be found, for example, in the works of Smith [176] and Willmott [191].

10.3.1 Improving the Convergence Properties

If only few time steps are chosen, we observe that the results oscillate strongly with the number of time steps used [176]. Nonetheless, as few as 20 time steps are already sufficient to obtain stable results for a plain vanilla option with a time to maturity of six months. But it is usually worthwhile to increase the number of time steps, depending on the accuracy required and the available computation time. Note however, that the improvement in the approximation resulting from increasing the number of time steps is limited. Improving the approximation requires a refinement of both the time-grid as well as the S-grid. The proportion of the number of t-points to the number of S-points in the grid depends on the lifetime of the option, the required width of the S-grid (see below) and the type of difference scheme applied. The Crank-Nicolson scheme, for example, requires a significantly coarser t-grid to attain comparable accuracy to that obtained from using the implicit method with the otherwise same parameters. Experimenting with these parameters is in any case recommended.

In general, it makes sense to start with a *uniform* grid. An analysis of the solution surface, i.e., the price function at each point of the grid quickly reveals information about the time regions at which new grid points should be introduced. Figure 10.1 shows such a solution surface for an American call option with discrete dividend payments (see also the next section) approximately half way through its lifetime. Additional time points have been introduced at and immediately before the due date of the dividend payment. Doing so exactly incorporates the dividend payment into the calculation. We can clearly see where exercising shortly before the dividend payment is optimal (the jump occurs as a result). Likewise, it makes sense to introduce additional grid points in the time region shortly before the maturity of the option since there the curvature of the solution surface is quite large.

At every time point where an external shock such as a dividend payment occurs, the addition of grid points prevents the oscillation of the solution dependent on the number of t-grid points. This also holds for the S-grid at the points where the payoff function is discontinuous. For example, it makes sense to choose the strike price as a grid point, and also the underlying value S

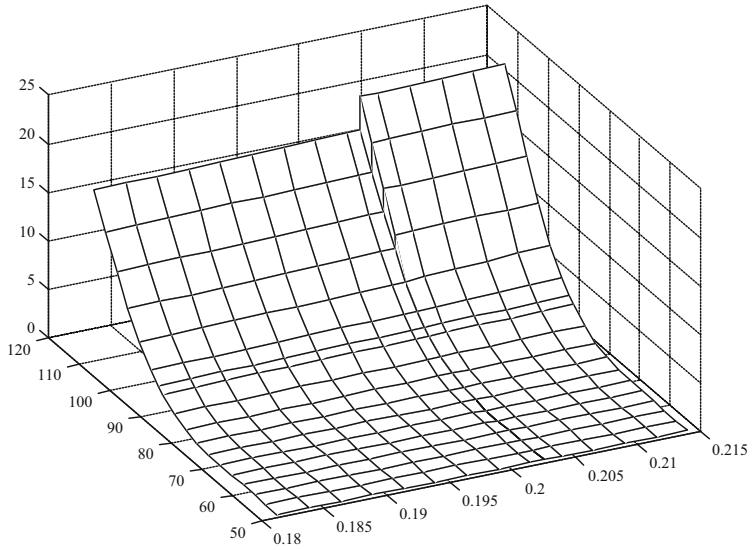

Fig. 10.1 Part of the solution surface of an American plain vanilla call on an underlying with a discrete dividend payment

for which the option price is to be determined. Likewise the barrier of a barrier option should ideally be directly on the grid. However, a non-uniform grid can *reduce* accuracy, since, for example, the Crank-Nicolson scheme is exact up to second order only for a *uniform* grid in t. In practice serious consideration should thus be giving as to where and whether additional grid points should be introduced.[7]

A good choice of whole S-region, i.e., of S_0 and S_{M+1} can substantially increase the accuracy attained. A rule of thumb is commonly used for choosing S_0 and S_{M+1}, such as

$$S_0 = \frac{1}{4}\min(S, K)$$
$$S_{M+1} = 4\max(S, K)\ ,$$

where here, S is the spot price of the underlying at time $t = 0$ and K the strike price of the option. The S-region thus obtained, however, (depending on the values of other parameters such as the volatility and the interest rate

[7] Experience shows that increasing/decreasing the step size in a regular manner works often well.

term structure) seldom leads to an optimally accuracy if the t-grid and the number of S-steps are assumed as given.

Alternatively, we could attempt to find the minimum (and maximum) value the underlying can attain in its lifetime *with a given probability*. This can be done if the underlying's cumulative probability distribution P has already been established. In the simplest cases, even an analytic formula for this probability can be derived (see Sect. A.4). In order to obtain minimal (maximal) S-values for a given probability (confidence) c, the expression $P(x \leq a) = c$ must be solved for a, the *percentile* of the underlying's distribution corresponding to the confidence level c. It is often worth the trouble to make this single calculation for each valuation to establish the S-region since it can result in a more precise solution with substantially fewer S-steps.

10.4 Discrete Dividends

Consideration of discrete dividends represents a further complication. The particular difficulty is that the different methods available rely on different assumptions. One commonly used method is based on the separation of the stochastic and deterministic components of the stochastic process S. The dividend payments correspond to the deterministic component. One assumption is thus that the exact value of the dividend payment is known a priori. The present value D_t of the dividends which are to be paid until maturity of the option is subtracted from the spot rate S of the underlying:

$$\widetilde{S} = S - D_t \ .$$

Instead of S we now use $\widetilde{S}$ as the new variable in the Black-Scholes equation. D_t is the nominal value of the dividends discounted back to t (where t denotes the time step currently under consideration). This process is consistent with the Black-Scholes formula for the pricing of a plain vanilla European option. There, discrete dividends are generally taken into consideration by subtracting the present value of the dividends from the spot price of the underlying. Consequently, the same volatility holds for both cases. Note however, that the *intrinsic* value of the option is still given by

$$\max(S - X, 0) = \max(\widetilde{S} + D_t - X, 0) \ .$$

This value is required to determine the free boundary condition for an early exercise of the option.

In the world of finite differences, we have an additional and fundamentally different method at our disposal. We first rewrite Eq. 7.8 replacing the dividend term $q(t)S$ resulting from the dividend rate $q(t)$ with $D(S,t)$:

$$\frac{\partial V}{\partial t} + (r(t)S - D(S,t))\frac{\partial V}{\partial S} + \frac{1}{2}\sigma(S,t)S^2\frac{\partial^2 V}{\partial S^2} = r(t)V \ .$$

The new term $D(S,t)$ represents the nominal value of the dividend payments at time t. If no dividends are paid at time t, the value of this function $D(S,t)$ is equal to zero. $D(S,t)$ is thus highly discontinuous. At time τ of a dividend payment, the option price must remain continuous, although the spot price of S is reduced by the amount given by the dividend payment D:

$$\lim_{\epsilon\to 0} V(S(\tau-\epsilon),\tau-\epsilon) = \lim_{\epsilon\to 0} V(S(\tau+\epsilon),\tau+\epsilon) \quad \epsilon > 0 \ ,$$

where

$$\lim_{\epsilon\to 0} S(t-\epsilon) = S$$
$$\lim_{\epsilon\to 0} S(t+\epsilon) = S - D \ .$$

This jump in S can be simulated with finite difference methods by first determining the solution vector at time τ. Subsequently, the vector is translated by D so that the continuity condition is satisfied. If necessary, missing intermediate values must be determined by interpolation, or in the case of boundary values, by extrapolation.

10.5 Example

The download website [50] accompanying this book contains a complete, executable example program written in Visual Basic (see FiniteDifferenceMethod.xlsm). The program was not written for optimal performance, but was structured in such a way as to incorporate all the concepts introduced in this section. For example, the S-grid is recalculated for every new time step (which, of course, need not be done if the grid is uniform). The volatility, interest rates and payoff are separated into individual functions to clearly organize the points at which such structures could be loaded externally via interfaces. If, for example, the interest rate or volatility is constant, this effort is superfluous. The program is structured to be, for the most part, self-

explanatory with helpful comments incorporated into the code. It can serve as a starting point for generating optimal-performance variants suitable for specific problems. This Excel workbook also demonstrates the calculation of option sensitivities, known as *Greeks*, which will be introduced in detail later in Chap. 12. Thus, this workbook can serve as a complete little option calculator (the yellow fields are input fields, a standard which applies to all workbooks on the accompanying website [50]).

In addition to the L-U decomposition introduced above, the example includes a further method for solving a system of equations, which is widely used in practice. The method is a very old procedure referred to as *Gaussian elimination*. Gaussian elimination is just as fast as the L-U decomposition and can be applied to both American and European options. The idea is even simpler than that of the L-U decomposition. Instead of decomposing the matrix in Eq. 10.15 into the matrix product of an upper and lower triangular matrix, it is transformed into a single triangular matrix with non-zero terms appearing only in the diagonal and a single off-diagonal. There are two possible ways of doing this: the matrix can be transformed into an upper or lower triangular matrix by eliminating the lower off-diagonal or upper off-diagonal terms respectively.

The first element in the lower off-diagonal, a_2, is eliminated by multiplying the first row of the matrix by a_2/b_1, and subtracting the result from the second row. After doing so, the second row contains only two elements, namely $b_2' = b_1 - c_1 a_2/b_1$ in diagonal and c_2 in the upper off-diagonal. This procedure is repeated analogously by multiplying the second row by a_3/b_2' and subtracting from the third, and so on. In terms of the system of equations, this means that certain equations are multiplied by constants and subtracted from one another. These are the usual operations implemented when solving a system of equations. In performing such operations, it is clear (in contrast to the L-U decomposition) that the right-hand side of the equation also changes. Carrying out this procedure for the entire matrix yields the following system of equations:

$$\begin{pmatrix} b_1' & c_1 & 0 & \cdots\cdots & 0 \\ 0 & b_2' & c_2 & 0 & \cdots \\ \vdots & 0 & b_3' & c_3 & \ddots & \cdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & \ddots & c_{M-2} \\ 0 & \cdots\cdots\cdots & & 0 & b_{M-1}' \end{pmatrix} \begin{pmatrix} W_{1,\,j} \\ W_{2,\,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,\,j} \end{pmatrix} = \begin{pmatrix} d_{1,\,j}' \\ d_{2,\,j}' \\ \vdots \\ \vdots \\ d_{M-2,\,j}' \\ d_{M-1,\,j}' \end{pmatrix}$$

where

$$b'_1 = b_1 \ , \quad d'_{1,\,j} = d_{1,j}$$
$$b'_i = b_i - c_{i-1}\frac{a_i}{b'_{i-1}} \ , \quad d_{i,\,j}{}' = d_{i,\,j} - d'_{i-1,\,j}\frac{a_i}{b'_{i-1}} \ , \quad \forall i = 2, \ldots, M-1 \ .$$

This system can be solved quite easily by starting with the row containing only one non-zero term, in this case the last row, and working upwards from the bottom. We obtain the option values

$$\widetilde{W}_{M-1,\,j} = \max\left[\frac{d'_{M-1,\,j}}{b'_{M-1}}, P(S_{M-1})\right]$$
$$\widetilde{W}_{i,\,j} = \max\left[\frac{d'_{i,\,j} - c_i\ \widetilde{W}_{i+1,\,j}}{b'_i}, P(S_i)\right] \quad \forall i = 1, \ldots, M-2 \ .$$

To this extent, this method offers an alternative to the L-U decomposition for the valuation of European options. The free boundary condition for American options is dealt with, as in the method of Lamberton and Lapeyre, by calculating the value of the option by Gaussian elimination as if it were European and, at each time step, comparing it with the intrinsic value of the option $P(S)$, i.e., the payoff profile. The value of the American option at this time point is then defined as the greater of the two. The components of the solution vector $\widetilde{W}_{i,j}$ are thus given by

$$\widetilde{W}_{M-1,\,j} = \max\left[\frac{d'_{M-1,\,j}}{b'_{M-1}}, P(S_{M-1})\right]$$
$$\widetilde{W}_{i,\,j} = \max\left[\frac{d'_{i,\,j} - c_i\ \widetilde{W}_{i+1,\,j}}{b'_i}, P(S_i)\right] \quad \forall i = 1, \ldots, M-2 \ .$$

As was already stressed earlier when we presented the method of *Lamberton and Lapeyre*, this procedure functions only when we calculates along the S-grid points starting from the exercise region, where the value of the option is given by its intrinsic value (the payoff profile), toward the region where the Black-Scholes equation holds. Otherwise, we would observe an edge or jump at the transition from one region into the other. The Gaussian elimination procedure described above starts from large S (index $i = M-1$) and calculates backwards

step by step towards smaller values of S. This works fine for call options since early exercise of a call can only (if at all) be optimal for large S because for small S (for $S < K$) the intrinsic value of the call is of course zero.

For put options, however, the exercise region can only lie in areas where the value of S is small, since for large S (for $S > K$), the intrinsic value of the put is zero, the option itself having only a time value. Therefore the second possibility for performing a Gaussian elimination has to be used for puts, namely eliminating the upper off-diagonal. This results in a procedure which begins with small S proceeding step by step towards larger S-values. The upper off-diagonal is eliminated by multiplying the last row of the matrix by c_{M-2}/b_{M-1}, subtracting the result from the next to last row. Proceeding analogously, the newly created next to last row is multiplied by c_{M-3}/b'_{M-2} and subtracted from the row lying immediately above it, and so on. Performing this procedure for the entire matrix yields the following system of equations:

$$\begin{pmatrix} b'_1 & 0 & \cdots & \cdots & \cdots & 0 \\ a_2 & b'_2 & 0 & & & \vdots \\ 0 & a_3 & b'_3 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & a_{M-2} & b'_{M-2} & 0 \\ 0 & \cdots & \cdots & 0 & a_{M-1} & b'_{M-1} \end{pmatrix} \begin{pmatrix} W_{1,\,j} \\ W_{2,\,j} \\ \vdots \\ \vdots \\ \vdots \\ W_{M-1,\,j} \end{pmatrix} = \begin{pmatrix} d'_{1,\,j} \\ d'_{2,\,j} \\ \vdots \\ \vdots \\ d'_{M-2,\,j} \\ d'_{M-1,\,j} \end{pmatrix}$$

where

$$b'_{M-1} = b_{M-1}\,, \quad d'_{M-1,\,j} = d_{M-1,\,j}$$

$$b'_i = b_i - a_{i+1}\frac{c_i}{b'_{i+1}}\,, \quad d'_{i,\,j} = d_{i,\,j} - d'_{i+1,\,j}\frac{c_i}{b'_{i+1}}\,, \quad \forall i = 1, \ldots, M-2\,.$$

This system can be solved quite easily by starting with the row containing only one non-zero element, in this case the first row, and proceeding from top to bottom. We thus obtain the option values

$$W_{1,\,j} = \frac{d'_{1,j}}{b'_1}$$

$$W_{i,\,j} = \frac{d'_{i,j} - a_i W_{i-1,j}}{b'_i} \quad \forall i = 2, \ldots, M-1\,.$$

The components of the solution vector $\widetilde{W}_{i,j}$ for an American option with payoff $P(S)$ is given by

$$\widetilde{W}_{1,\,j} = \max\left[\frac{d'_{1,j}}{b'_1},\, P(S_1)\right]$$

$$\widetilde{W}_{i,\,j} = \max\left[\frac{d'_{i,j} - a_i\,\widetilde{W}_{i-1,\,j}}{b'_i},\, P(S_i)\right] \quad \forall i = 2, \ldots, M-1\ .$$

10.5.1 Relationship Between Explicit Finite Difference and Tree Methods

Numerically, the trinomial tree and the explicit method of finite differences are equivalent, at least in certain special cases. Consider the ith node of the time step j. This corresponds to an underlying price S_{ji}. This node is connected to the nodes $i+1$, i and $i-1$ of the time step $j+1$. The value of the option V_{ji} at node i of the time step j is given by

$$V_{ji} = B(t_j, t_{j+1})\left[p^+ V_{j+1,i+1} + p^0 V_{j+1,i} + p^- V_{j+1,i-1}\right]\ .$$

The analogous expression for the explicit finite difference method is given by Eq. 10.22 with the coefficients given by Eq. 10.12 (or Eq. 10.10 for uniform grids):

$$W_{i,j} = \left(1 - (t_{j+1} - t_j)B_{i,j+1}\right) W_{i,\,j+1} - (t_{j+1} - t_j)A_{i,j+1}\, W_{i-1,j+1} - (t_{j+1} - t_j)C_{i,j+1}\, W_{i+1,j+1}\ .$$

We immediately see the similarity in structure. The probabilities can be easily associated with the coefficients of $W_{i+1,j+1}$, $W_{i,j+1}$ and $W_{i-1,j+1}$. This requires the grid points of the finite difference grid to correspond exactly to the nodes of the trinomial tree and thus the restriction $S_{ji} = S_{j+1,i}$ to hold for the nodes of the trinomial tree. The finite difference grid was constructed specifically to satisfy just this very condition (even for non-uniform grids) to prevent the expressions for the finite differences from becoming unnecessarily complicated. The trinomial tree appears more flexible from this point of view. However, it has all the disadvantages associated with the explicit method. Moreover, a procedure corresponding to the implicit or Crank-Nicolson method is not available for trinomial trees.