

Derivative Pricing, Numerical Methods

Numerical methods are needed for derivatives pricing in cases where analytic solutions are either unavailable or not easily computable. Examples of the former case include American-style options and most discretely observed path-dependent options. Examples of the latter type include the analytic formula for valuing continuously observed Asian options in [28], which is very hard to calculate for a wide range of parameter values often encountered in practice, and the closed form solution for the price of a discretely observed partial barrier option in [32], which requires high dimensional numerical integration.

The subject of numerical methods in the area of derivatives valuation and **hedging** is very broad. A wide range of different types of contracts are available, and in many cases there are several candidate models for the stochastic evolution of the underlying state variables. Many subtle numerical issues can arise in various contexts. A complete description of these would be very lengthy, so here we will only give a sample of the issues involved.

Our plan is to first present a detailed description of the different methods available in the context of the **Black–Scholes–Merton** [6, 43] model for simple European and American-style equity options. There are basically two types of numerical methods: techniques used to solve partial differential equations (PDEs) (these include the popular binomial and trinomial tree approaches), and Monte Carlo methods. We will then describe how the methods can be adapted to more general contexts, such as derivatives dependent on more than one underlying factor, path-dependent derivatives, and derivatives with discontinuous payoff functions. Familiarity with the basic theory of derivative pricing will generally be assumed, though some of the basic concepts will be reviewed when we discuss the binomial method. Readers seeking further information about these topics should consult texts such as [33, 56], which are also useful basic references for numerical methods.

To set the stage, consider the case of a nondividend paying stock with a price S , which evolves according to geometric Brownian motion

$$dS = \mu S dt + \sigma S dB, \quad (1)$$

where the drift rate μ and volatility σ are assumed to be constants, and B is a standard **Brownian motion**. Following the standard no-arbitrage analysis, the value V of a derivative claim on S may be found by solving the partial differential equation

$$\frac{1}{2}\sigma^2 S^2 V_{SS} + rSV_S + V_t - rV = 0, \quad (2)$$

subject to appropriate boundary conditions. Note that in (2), subscripts indicate partial derivatives, r is the continuously compounded risk free interest rate, and t denotes time. As (2) is solved backward from the maturity date of the derivative contract T to the present, it is convenient to rewrite it in the form

$$V_\tau = \frac{1}{2}\sigma^2 S^2 V_{SS} + rSV_S - rV, \quad (3)$$

where $\tau = T - t$. The terminal payoff of the contract becomes an initial condition for (3). For instance, a European call option with an exercise price of K has a value at expiry of $V(S, t = T) = V(S, \tau = 0) = \max(S_T - K, 0)$, where S_T is the underlying stock price at time T . Similarly, a European put option would have a payoff condition of $V(S, t = T) = \max(K - S_T, 0)$. For these simple examples, there is no need for a numerical solution because the Black–Scholes formula offers a well-known and easily computed analytic solution. Nonetheless, these problems provide useful checks for the performance of numerical schemes.

It is also important to note that we can solve (2) by calculating an expectation. This follows from the *Feynman–Kac solution* (see [23] for more details)

$$V(S, t) = \exp(-r(T - t)) E^*(V(S, T)), \quad (4)$$

where the expectation operator E is superscripted with $*$ to indicate that we are calculating the expected terminal payoff of the contract not under the process (1), but under a modified version of it, where μ is replaced with the risk free interest rate r . Formulation (4) is the basis for Monte Carlo methods of option valuation.

The Binomial Method

Perhaps the easiest numerical method to understand is the **binomial method** that was originally developed in [21]. We begin with the simplest model of uncertainty; there is one period, and the underlying

2 Derivative Pricing, Numerical Methods

stock price can take on one of two possible values at the end of this period. Let the current stock price be S , and the two possible values after one period be Su and Sd , where u and d are constants with $u > d$. Denote one plus the risk free interest rate for the period as R . Absence of arbitrage requires that $u > R > d$ (if $u \leq R$, then the stock would be dominated by a risk free bond; if $R < d$, then the stock would dominate the risk free bond). Figure 1 provides an illustration of this model.

We know the value of the option at expiry as a function of S . Suppose we are valuing a European call option with strike price K . Then the payoff is $V_u = \max(Su - K, 0)$, if $S = Su$ at T and $V_d = \max(Sd - K, 0)$, if $S = Sd$ at that time. Consider an **investment strategy** involving the underlying stock and a risk free bond that is designed to replicate the option's payoff. In particular, suppose we buy h shares of stock and invest C in the risk free asset. We pick h and C to satisfy

$$\begin{aligned} hSu + RC &= V_u \\ hSd + RC &= V_d. \end{aligned} \quad (5)$$

This just involves solving two linear equations in two unknowns. The solution is

$$\begin{aligned} h &= \frac{V_u - V_d}{Su - Sd} \\ C &= \frac{V_d Su - V_u Sd}{R(Su - Sd)}. \end{aligned} \quad (6)$$

The cost of entering into this strategy at the start of the period is $hS + C$. No-arbitrage considerations then tell us that the price of the option today must equal the cost of this alternative investment strategy that produces the same future payoffs as the option.

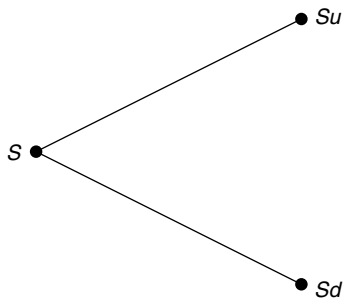


Figure 1 A one-period binomial model

In other words,

$$\begin{aligned} V &= \frac{V_u - V_d}{S(u - d)} S + \frac{S(V_d u - V_u d)}{RS(u - d)} \\ &= \frac{V_u - V_d}{u - d} + \frac{V_d u - V_u d}{R(u - d)} \\ &= \frac{1}{R} \left[\left(\frac{R - d}{u - d} \right) V_u + \left(\frac{u - R}{u - d} \right) V_d \right] \\ &= \frac{1}{R} [\pi V_u + (1 - \pi) V_d], \end{aligned} \quad (7)$$

where $\pi = (R - d)/(u - d)$. Note that the no-arbitrage restriction $u > R > d$ implies that $0 < \pi < 1$, so we can interpret π (and $1 - \pi$) as probabilities, giving the intuitively appealing notion of discounting expected future payoffs to arrive at today's value. If these probabilities were the actual probabilities of the stock price movement over the period, then the expected rate of return on the stock would be $\pi u + (1 - \pi)d = R$. So (7) indicates that we calculate derivative values by assuming that the underlying asset has an expected growth rate equal to the risk free interest rate, calculating expected option payoffs, and then discounting at the risk free interest rate. The correspondence with (4) is obvious.

Although the model above is clearly far too simple to be of any practical relevance, it can easily be extended to a very useful scheme, simply by dividing the derivative contract life into various subperiods. In each subperiod, the underlying asset will be allowed to move to one of two possible values. This will allow us to construct a tree or lattice. With n subperiods, we will have a total of 2^n possible stock price paths through the tree. In order to keep things manageable, we will assume that $d = 1/u$. Then a u move followed by a d move will lead to the same stock price as two moves in the opposite sequence. This ultimately gives $n + 1$ terminal stock prices, vastly improving the efficiency of the numerical scheme (in fact, without a recombining tree, constraints on computer memory may make it impossible to implement a binomial method with a realistic number of subperiods). Figure 2 provides an illustration for the case of $n = 3$ subperiods. There are four terminal stock prices, but eight possible paths in the tree. (The nodes Su^3 and Sd^3 can each only be arrived at by a single path, with either three consecutive u moves or three consecutive d moves. The node Su^2d can be reached in three ways, each involving two u moves and one d

move. Similarly, three paths lead to the node Sud^2 .) As we move through the tree, the implied hedging strategy (6) will change, requiring a rebalance of the replicating portfolio. As it turns out, the replicating portfolio is *self-financing*, meaning that any changes in its composition are financed internally. The entire cost of following the strategy comes from its initial setup. Thus we are assured that the price of the option is indeed the initial cost of the replication strategy.

The binomial method can easily be programmed to contain hundreds (if not thousands) of subperiods. The initial option value is calculated by rolling backwards through the tree from the known terminal payoffs. Every node prior to the end leads to two successor nodes, so we are simply repeating the one period model over and over again. We just apply (7) at every node.

In the limit as $n \rightarrow \infty$, we can converge to the Black–Scholes solution in continuous time. Let r be the continuously compounded risk free interest rate, σ be the annual standard deviation of the stock price,

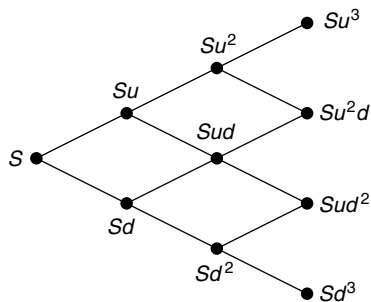


Figure 2 A three period binomial model. Note that there are four terminal stock prices, but eight possible paths through the tree

and let Δt be the time interval for each step (e.g. with $T = 1$ year and $n = 50$, $\Delta t = 0.02$). We will converge to the Black–Scholes price as $\Delta t \rightarrow 0$ (that is, as $n \rightarrow \infty$), if we set $u = e^{\sigma\sqrt{\Delta t}}$, $d = 1/u$, and $\pi = (e^{r\Delta t} - d)/(u - d)$.

In addition to the option value, we are frequently interested in estimating the ‘Greeks’. These are hedging parameters such as delta ($\partial V/\partial S$) and gamma ($\partial^2 V/\partial S^2$). In the context of the binomial method, these can be estimated by simply taking numerical derivatives of option values in the stock price tree. One simple trick often used to improve accuracy involves taking a couple of extra timesteps (backwards beyond the starting date), so that more than one value is available at the timestep corresponding to the initial time. This permits these derivatives to be evaluated at that time, rather than one or two timesteps later.

The binomial method is easily adaptable to the case of American options. As we roll back through the tree, we simply replace the computed value from above (which is the value of holding onto the option for another timestep) with the maximum of this and its payoff if it were to be exercised rather than held. This provides a very simple and efficient algorithm. Table 1 presents an illustrative example for European and American put options with $S = K = 100$, $r = 0.08$, $T = 1$, and $\sigma = 0.30$. The Black–Scholes value for the European put is 8.0229, so in this case we get a quite accurate answer with at most a few hundred timesteps. The rate of convergence for this type of numerical method may be assessed by repeatedly doubling the number of timesteps (and hence the number of grid points), and calculating the ratio of successive changes in the computed solution value. This will be around two if convergence is at a first order or linear rate, and approximately four if

Table 1 Illustrative results for the binomial method. The parameters are $S = K = 100$, $r = 0.08$, $T = 1$ year, $\sigma = 0.30$. The exact value for the European put is 8.0229. Value is the computed option value. Change is the difference between successive option values as the number of timesteps (and thus grid size) is doubled. Ratio is the ratio of successive changes

No. of timesteps	European put			American put		
	Value	Change	Ratio	Value	Change	Ratio
50	7.9639	n.a.	n.a.	8.8787	n.a.	n.a.
100	7.9934	0.0295	n.a.	8.8920	0.0133	n.a.
200	8.0082	0.0148	1.99	8.8983	0.0063	2.11
400	8.0156	0.0074	2.00	8.9013	0.0030	2.10
800	8.0192	0.0036	2.06	8.9028	0.0015	2.00
1600	8.0211	0.0019	1.89	8.9035	0.0007	2.14

convergence is at a second order or quadratic rate. It is well-known that the standard binomial model exhibits linear convergence. This is borne out in Table 1 for both the European and American cases.

The ease of implementation and conceptual simplicity of the binomial method account for its enduring popularity. It is very hard to find a more efficient algorithm for simple options. Numerous researchers have extended it to more complicated settings. However, as we shall see below, this is often not an ideal strategy.

Numerical PDE Methods

A more general approach is to solve (3) using a numerical PDE approach. There are several different possibilities here including finite differences, finite elements, and finite volume methods. In a one-dimensional setting, the differences between these various techniques are slight; in many cases all of these approaches give rise to the same set of discrete equations and the same option values.

We will consider the finite difference approach here as it is the easiest to describe. The basic idea is to replace the derivatives in (3) with discrete differences calculated on a finite grid of values for S . This is given by S_i , $i = 0, \dots, p$, where $S_0 = 0$ and $S_p = S_{\max}$, the highest value for the stock price on the grid. Boundary conditions will depend on the contract, and can be arrived at using obvious financial reasoning in most cases. For instance, for a European call option at S_{\max} , we can use $V(S_p) = S - Ke^{-r\tau}$, reflecting the virtually certain exercise of the instrument at expiry. Similarly, a put option is almost surely not going to be exercised if S is very large, so we would use $V(S_p) = 0$ in this situation. At $S_0 = 0$, a European put option would be worth the present value of the exercise price, while a call would be worth zero. We can specify boundary conditions using these values, but it is not required. This is because at $S = 0$, the PDE (3) reduces to the easily solved ordinary differential equation $V_\tau = -rV$.

In most finance references, an equally spaced grid is assumed but this is not necessary for anything other than ease of exposition. In the following, we will consider the more general case of a nonuniform grid since it offers considerable advantages in some contexts. For the moment, let us consider the derivatives with respect to S . Let V_i denote the option value at

grid node i . Using a first-order Taylor's series expansion, we can approximate V_S at node i by either the forward difference

$$V_S \approx \frac{V_{i+1} - V_i}{S_{i+1} - S_i}, \quad (8)$$

or the backward difference

$$V_S \approx \frac{V_i - V_{i-1}}{S_i - S_{i-1}}, \quad (9)$$

where \approx serves to remind us that we are ignoring higher-order terms. Defining $\Delta S_{i+1} = S_{i+1} - S_i$ (and similarly for ΔS_i), we can form a central difference approximation by combining (8) and (9) to get

$$V_S \approx \frac{V_{i+1} - V_{i-1}}{\Delta S_{i+1} + \Delta S_i}. \quad (10)$$

In the case where $\Delta S_i = \Delta S_{i+1}$, this is a second-order approximation. (It is possible to derive a more complicated expression which is second order correct for an unevenly spaced grid, but this is not used much in practice because grid spacing typically changes fairly gradually, so there is not much cost to using the simpler expression (10).) Similarly, using higher-order expansions and some tedious manipulations, we can derive the following approximation for the second derivative with respect to the stock price at node i

$$V_{SS} \approx \frac{\frac{V_{i+1} - V_i}{\Delta S_{i+1}} + \frac{V_{i-1} - V_i}{\Delta S_i}}{\frac{\Delta S_{i+1} + \Delta S_i}{2}}. \quad (11)$$

This is also only first-order correct, but for an equally spaced grid with $\Delta S_i = \Delta S_{i+1} = \Delta S$, it reduces to

$$V_{SS} \approx \frac{V_{i+1} - 2V_i + V_{i-1}}{(\Delta S)^2}, \quad (12)$$

which is second-order accurate. We will also use a first-order difference in time to approximate V_τ . Letting V_i^n denote the option value at node i and time level n , we have (at node i)

$$V_\tau \approx \frac{V_i^{n+1} - V_i^n}{\Delta \tau}, \quad (13)$$

where $\Delta \tau$ is the discrete timestep size. Note that hedging parameters such as delta and gamma can be easily estimated using numerical differentiation of the computed option values on the grid.

Different methods arise depending upon the time level at which the spatial derivatives are evaluated.

Note that we are seeking to advance the discrete solution forward from time n to $n + 1$. (Recall that we are using the forward equation (3), so time is actually running backwards.) If we evaluate these derivatives at the old time level n , we have an *explicit* method. In this case, plugging in our various approximations to the PDE, we obtain

$$\frac{V_i^{n+1} - V_i^n}{\Delta\tau} = \frac{\sigma^2}{2} S_i^2 \left[\frac{\frac{V_{i+1}^n - V_i^n}{\Delta S_{i+1}} + \frac{V_{i-1}^n - V_i^n}{\Delta S_i}}{\frac{\Delta S_{i+1} + \Delta S_i}{2}} \right] + r S_i \left[\frac{V_{i+1}^n - V_{i-1}^n}{\Delta S_{i+1} + \Delta S_i} \right] - r V_i^n. \quad (14)$$

This can be rearranged to get

$$V_i^{n+1} = \alpha_i \Delta\tau V_{i-1}^n + [1 - (\alpha_i + \beta_i + r) \Delta\tau] V_i^n + \beta_i \Delta\tau V_{i+1}^n, \quad (15)$$

where

$$\alpha_i = \frac{\sigma^2 S_i^2}{\Delta S_i (\Delta S_{i+1} + \Delta S_i)} - \frac{r S_i}{\Delta S_{i+1} + \Delta S_i}, \quad (16)$$

$$\beta_i = \frac{\sigma^2 S_i^2}{\Delta S_{i+1} (\Delta S_{i+1} + \Delta S_i)} + \frac{r S_i}{\Delta S_{i+1} + \Delta S_i}. \quad (17)$$

Similarly, we can derive an *implicit* method by evaluating the spatial derivatives at the new time level $n + 1$. Skipping over the details, we arrive at

$$-\alpha_i V_{i-1}^{n+1} + [1 + (\alpha_i + \beta_i + r) \Delta\tau] V_i^{n+1} - \beta_i V_{i+1}^{n+1} = V_i^n. \quad (18)$$

Using our ordinary differential equation $V_\tau = -rV$ at $S = 0$ and an appropriately specified boundary condition $V_p = V(S_p)$ at S_{\max} , we can combine (15) and (18) into a compact expression. In particular, let V^n be a column vector containing the elements of the solution at time n , that is, $V^n = [V_0^n, V_1^n, \dots, V_p^n]'$. Also, let the matrix, A , be given by

$$A = \begin{bmatrix} r & 0 & 0 & 0 & 0 & \dots & 0 \\ -\alpha_2 & (r + \alpha_2 + \beta_2) & -\beta_2 & 0 & 0 & \dots & 0 \\ 0 & -\alpha_3 & (r + \alpha_3 + \beta_3) & -\beta_3 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & -\alpha_{p-1} & (r + \alpha_{p-1} + \beta_{p-1}) & -\beta_{p-1} \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (19)$$

Then we can write

$$(I + \theta A \Delta\tau) V^{n+1} = (I - (1 - \theta) A \Delta\tau) V^n, \quad (20)$$

where I is the identity matrix. Note that $\theta = 0$ is an explicit method, while $\theta = 1$ gives an implicit scheme. Either of these approaches is first-order correct in time. A second-order scheme can be obtained by specifying $\theta = 1/2$. This is known as a *Crank–Nicolson* method.

It is worth noting that what the literature refers to as trinomial trees are effectively just explicit finite difference methods. (Trinomial trees are similar to the binomial method, except that the stock price can branch to any of three, not two, successor nodes. To be precise, there is a slight difference between the explicit finite difference method described here and the traditional trinomial tree. This arises from treating the rV term in the PDE (3) in an implicit fashion.) Although this is well known, what is less commonly mentioned in the literature is that we can also view the binomial method as a particular explicit finite difference method using a log-transformed version of (3) (see [31]). Therefore, all of the observations below regarding explicit type methods also apply to the binomial and trinomial trees that are pervasive in the literature.

It is critical that any numerical method be *stable*. Otherwise, small errors in the computed solution can grow exponentially, rendering it useless. Either a Crank–Nicolson or an implicit scheme is unconditionally stable, but an explicit method is stable if and only if

$$\Delta\tau \leq \min_i \left[\frac{1}{\alpha_i + \beta_i + r} \right]. \quad (21)$$

In the case of a uniformly spaced grid where $\Delta S_{i+1} = \Delta S_i = \Delta S$, (21) simplifies to

$$\Delta\tau \leq \min_i \frac{(\Delta S)^2}{\sigma^2 S_i^2 + r(\Delta S)^2}. \quad (22)$$

This indicates that the timestep size must be very small if the grid spacing is fine. This can be a very severe limitation in practice because of the following reasons:

- It can be very inefficient for valuing long-term derivative contracts. Because of the parabolic nature of the PDE (3), the solution tends to smooth out as we proceed further in time, so we would want to be taking larger timesteps.
- A fine grid spacing is needed in cases where the option value has discontinuities (such as digital options or barrier options). This can lead to a prohibitively small timestep.

By contrast, neither Crank–Nicolson nor implicit methods are subject to this stability restriction. This gives them the potential for considerably more flexibility in terms of grid design and timestep size as compared to explicit methods.

Boundary conditions are another consideration. For explicit methods, there is no need to specify them because we can solve backwards in time in a triangular region (much like the binomial method, but with three branches). Some authors claim this is a significant advantage for explicit methods [34]. Balanced against this, however, is the fact that pricing problems of long enough maturity eventually lead to an excessive number of nodes (because we do not impose a boundary condition at S_{\max} , the highest stock price in the tree keeps on growing).

This does, however, lead to the issue of how large S_{\max} needs to be. Various rules-of-thumb are cited in the literature, such as three or four times the exercise price. A call option will typically require a larger value than a put, since for a call the payoff function has considerable value at S_{\max} , while it is zero for a put. In general, it can be shown that S_{\max} should be proportional to $\sigma\sqrt{T}$, so options either on underlying assets with high volatilities, or with long times until expiry, should have higher values of S_{\max} .

Another issue concerns the solution of (20). Explicit methods ($\theta = 0$) determine V_i^{n+1} as a linear combination of three nodes at the preceding timestep, so there is no need to solve a set of simultaneous linear equations. This makes them easier to use than either Crank–Nicolson or implicit methods, which require solving a set of linear equations. However, A is a tridiagonal matrix (the only nonzero entries are on the diagonal, and on each adjacent side of it in each row). This means that the system can be solved

very quickly and efficiently using algorithms such as described in [46].

Another topic worth noting relates to the central difference approximation of V_S in (10). If this approximation is used, it is possible that α_i in (16) can become negative. This can cause undesirable effects such as spurious oscillations in our numerical solution. Even if such oscillations are all but invisible in the solution profile for the option price, they can be quite severe for the option delta V_S and gamma V_{SS} , which are important parameters for implementing hedging strategies. In the case of an implicit method, we can guarantee that oscillations will not happen by switching to the forward difference (8) at node i . This implies replacing (16) and (17) with

$$\alpha_i = \frac{\sigma^2 S_i^2}{\Delta S_i (\Delta S_{i+1} + \Delta S_i)}, \quad (23)$$

$$\beta_i = \frac{\sigma^2 S_i^2}{\Delta S_{i+1} (\Delta S_{i+1} + \Delta S_i)} + \frac{r S_i}{\Delta S_{i+1}}. \quad (24)$$

Although this is only first-order correct, the effects on the accuracy of our numerical solution are generally negligible. This is because the only nodes where a switch to forward differencing is required are remote from the region of interest, at least across a broad range of typical parameter values. For instance, suppose we use a uniform grid with $\Delta S_{i+1} = \Delta S_i = \Delta S$, and $S_i = i \Delta S$. Then, α_i in (16) will be nonnegative provided that $\sigma^2 \geq r/i$. Note that the case $i = 0$ does not apply since that is the boundary where $S = 0$. If $r = 0.06$ and $\sigma = 0.30$, then we will never switch to forward differences. If $r = 0.12$ and $\sigma = 0.15$, then we will move to forward differences for $i \leq 5$, that is, only for the first five interior nodes near the lower boundary. Note, however, that this may not be true in other contexts, such as interest rate or stochastic volatility models featuring mean-reversion.

What about an explicit method? It turns out that the stability restriction implies that $\alpha_i \geq 0$, so this is not an issue for this type of approach. However, for a Crank–Nicolson scheme, things are unfortunately rather ambiguous. As noted in [61], we can only guarantee that spurious oscillations will not occur if we restrict the timestep size to twice the maximum stable explicit timestep size. If we do not do this, then oscillations *may* occur, but we cannot tell in advance. However, they will be most prone to happen in situations where the solution is changing rapidly, such as near a discretely observed barrier, or close

to the strike price of a digital option. As will be discussed below, there are ways of mitigating the chance that oscillations will occur.

We have noted that an advantage of Crank–Nicolson and implicit methods over explicit methods is the absence of a timestep size restriction for stability. This permits the use of unevenly spaced grids that can provide significant efficiency gains in certain contexts. It also allows us to choose timestep sizes in a more flexible fashion, which offers some further benefits. First, it is straightforward to change the timestep size, so that timesteps can be aligned with intermediate cash flow dates, arising from sources such as dividend payments made by the underlying stock or coupons from a bond. Second, values of long-term derivatives can be computed much more efficiently if we exploit the ability to take much larger timesteps as we move away from the expiry time. Third, it is not generally possible to achieve second-order convergence for American options using constant timesteps [25]. While it is possible to alter the timestep suitably in an explicit scheme, the tight

coupling of the grid spacing and timestep size due to the stability restriction makes it considerably more complicated. Some examples of this may be found in [24].

A simple, automatic, and quite effective method for choosing timestep sizes is described in [37]. Given an initial timestep size, a new timestep is selected according to the relative change in the option price over the previous timestep at various points on the S grid. Of course, we also want to ensure that timesteps are chosen to match cash flow dates, or observation intervals for discretely monitored path-dependent options. One caveat with the use of a timestep selector involves discrete barrier options. As the solution changes extremely rapidly near such barriers, chosen timesteps will be excessively small, unless we limit our checking to values of S that are not very close to a barrier.

To illustrate the performance of these various methods in a simple case, we return to our earlier example of a European put with $S = K = 100$, $r = 0.08$, $T = 1$, and $\sigma = 0.30$. Table 2 presents

Table 2 Illustrative results for numerical PDE methods for a European put option. The parameters are $S = K = 100$, $r = 0.08$, $T = 1$ year, $\sigma = 0.30$. The exact value is 8.0229. Value is the computed option value. Change is the difference between successive option values as the number of timesteps and the number of grid points is increased. Ratio is the ratio of successive changes

No. of grid nodes	No. of timesteps	Value	Change	Ratio
Explicit				
51	217	8.0037	n.a.	n.a.
101	833	8.0181	0.0144	n.a.
201	3565	8.0217	0.0036	4.00
401	14 329	8.0226	0.0009	4.00
Implicit				
51	50	7.9693	n.a.	n.a.
101	100	8.0026	0.0333	n.a.
201	200	8.0144	0.0118	2.82
401	400	8.0191	0.0047	2.51
801	800	8.0211	0.0020	2.35
1601	1600	8.0220	0.0009	2.22
Crank–Nicolson				
51	50	7.9974	n.a.	n.a.
101	100	8.0166	0.0192	n.a.
201	200	8.0213	0.0047	4.09
401	400	8.0225	0.0012	3.92
801	800	8.0228	0.0030	4.00
1601	1600	8.0229	0.0010	3.00

results for explicit, implicit, and Crank–Nicolson methods. Since in this simple case there is not much of an advantage to using nonuniform grid spacing or uneven timestep sizes, we simply start with an evenly spaced grid running from $S = 0$ to $S = 200$. We then refine this grid a number of times, each time by just inserting a new node halfway between previously existing nodes. For the implicit and Crank–Nicolson cases, the number of timesteps is initially set at 50 and doubled with each grid refinement. The number of timesteps in the explicit case cannot be determined in this way. In this example, we use the largest stable explicit timestep size. In order to satisfy the stability criterion (21), the number of timesteps must increase by roughly a factor of 4 with each grid refinement. This implies an increase in computational work of a factor of 8 with each refinement (versus 4 in the cases of both implicit and Crank–Nicolson). This limits our ability to refine the grid for this scheme, in that computational time becomes excessive. (This is obviously somewhat dependent on the particular implementation of the algorithm. In this case, no particular effort was made to optimize the performance of any of the methods. It is worth noting, however, that the explicit method with 401 grid points (and 14 329 timesteps) requires almost triple the computational time as the implicit method with 1601 grid points and 1600 timesteps.) It also means that the apparent quadratic convergence indicated in Table 2 for the explicit case is actually misleading (the value of 4 in the ‘Ratio’ column would demonstrate second-order convergence if the number of timesteps was being doubled, not quadrupled). All of the methods do converge, however, with the rate for the implicit method being approximately linear and that for Crank–Nicolson being approximately quadratic. For this particular example, time truncation error appears to be more significant than spatial truncation error. Notice how far the implicit method is away from the other two methods on the coarsest spatial grid. The explicit method does substantially better on this grid because it takes more than four times as many timesteps, while Crank–Nicolson does much better with the same number of timesteps as implicit because it is second-order accurate in time.

It must be said, however, that for this simple example, there is no compelling reason to use any of these methods over the binomial method. Comparing Tables 1 and 2, we observe that the binomial method requires 400 timesteps (and thus 401 terminal grid

nodes) to achieve an absolute error of 0.01 (for the European case), while the explicit method requires 101 nodes (and 833 timesteps). This level of accuracy is attained by the implicit method with 201 nodes and 200 timesteps, and by the Crank–Nicolson method with half as many nodes and timesteps. But this ignores computational time and coding effort. It also ignores the fact that we only calculate the option value for a single value of S with the binomial method, whereas we compute it across the entire grid for the other techniques. Roughly speaking, the binomial method takes about the same amount of CPU time as the implicit method to achieve an error within 0.01. The Crank–Nicolson method takes around 75% as much CPU time as these methods, while the explicit technique takes about twice as long. If we are only interested in the option value for a particular value on the S grid, then the binomial method performs quite well, and it has the virtue of requiring far less effort to implement.

Monte Carlo Methods

Monte Carlo methods are based on the Feynman–Kac solution (4). Originally introduced to the option pricing literature in [7], they have found wide applicability ever since. An excellent survey may be found in [9], and we shall rely heavily on it here. The basic idea is as follows. Recall the stochastic differential equation (1), which models the evolution of the underlying stock price over time. (Also recall that for derivative valuation, we replace the real drift rate μ with the risk free rate r .) We can simulate this evolution forward in time using the approximation

$$S(t + \Delta t) = S(t) + rS(t)\Delta t + \sigma S(t)\sqrt{\Delta t}Z, \quad (25)$$

where $Z \sim N(0, 1)$. This gives us values of S at $t = 0, \Delta t, 2\Delta t$, etc. We carry out this procedure until we reach the expiry time of the option contract T . Applying the payoff function to this terminal value for S gives us one possible realization for the ultimate value of the option. Now imagine repeating this many times, building up a distribution of option values at T . Then, according to (4), we can simply average these values and discount back to today, at the risk free interest rate, to compute the current value of the option. Hedging parameters such as delta and gamma may be calculated using techniques described in [14].

It should be noted that for the simple case of geometric Brownian motion (1), we can integrate to get the exact solution

$$S(T) = S(0) \exp \left[\left(\frac{r - \sigma^2}{2} \right) T + \sigma \sqrt{T} Z \right], \quad (26)$$

where once again $Z \sim N(0, 1)$. This avoids the error due to the first-order approximation (25). (There are higher-order approximation schemes, but these are not commonly used because they involve considerably higher computational overhead. See, for example [40], for a thorough discussion of these methods.) Unfortunately, exact expressions like (26) are rare.

Monte Carlo simulations produce a statistical estimate of the option value. Associated with it is a standard error that tends to zero at the rate of $M^{-1/2}$, where M is the number of independent simulation runs. Increasing the number of runs by a factor of 10 will thus reduce the standard error by a factor of a little more than three. An approximate 95% confidence interval for the option value can be constructed by taking the estimated price plus or minus two standard errors. Observe that when we cannot integrate the stochastic differential equation exactly (that is, we must step forward through time as in (25)), there is an additional approximation error. This is often overlooked, so standard errors in these cases are biased downwards. (Of course, by reducing the time interval Δt in (25), we can reduce this error, but what is commonly done in practice is to fix Δt at some interval such as one day and then to keep increasing M .)

In order to obtain more precise estimates, an alternative to simply doing more simulations is to use a *variance reduction technique*. There are several possibilities, the simplest being *antithetic variates*. This exploits the fact that if $Z \sim N(0, 1)$, then $-Z \sim N(0, 1)$. In particular, if $V(S_T^i)e^{-rT}$ denotes the discounted terminal payoff of the contract for the i th simulation run when S_T^i is evaluated using Z_i , then the standard Monte Carlo estimator of the option value is

$$\widehat{V} = \frac{1}{M} \sum_{i=1}^M V(S_T^i)e^{-rT}. \quad (27)$$

Now let \tilde{S}_T^i be the terminal value of the simulated stock price based on $-Z_i$. Then the estimated option

value using antithetic variates is

$$\widehat{V}_{AV} = \frac{1}{M} \sum_{i=1}^M \frac{V(S_T^i)e^{-rT} + V(\tilde{S}_T^i)e^{-rT}}{2}. \quad (28)$$

To calculate the standard error using this method, observe that Z_i and $-Z_i$ are not independent. Therefore the sample standard deviation of the M values of $[V(S_T^i)e^{-rT} + V(\tilde{S}_T^i)e^{-rT}]/2$ should be used.

Another method to drive down the variance is the *control variate* technique. Here we draw on the description in [9], which is more general than typically provided in references. The basic idea is as follows. Suppose there is another related option pricing problem for which we know the exact answer. (For example, consider the case in [7] in which the solution for an option on a stock that does not pay dividends is used as a control variate for valuing an option on a stock that does pay dividends.) Denote the exact value of the related problem by W , and its estimated value using Monte Carlo as \widehat{W} . Then the control variate estimator of the option value of interest is

$$\widehat{V}_{CV} = \widehat{V} + \beta(W - \widehat{W}). \quad (29)$$

The intuition is simple: we adjust our estimated price from standard Monte Carlo using the difference between the known exact value and its Monte Carlo estimate. As noted in [9], it is often assumed that $\beta = 1$, but that is not necessarily an optimal, or even good, choice. To minimize variance, the optimal value is

$$\beta^* = \frac{\text{Cov}(\widehat{V}, \widehat{W})}{\text{Var}(\widehat{W})}. \quad (30)$$

This can be estimated via linear regression using an additional prior set of independent simulations. Observe that the common choice of $\beta = 1$ implicitly assumes perfect positive correlation.

As a third possibility, importance sampling can sometimes be used to dramatically reduce variance. The intuition is easiest for pricing out-of-the-money options. Consider the example of an out-of-the-money put, with the current stock price above the strike. Since we simulate forward with a positive drift, most of the simulated stock price paths will finish out-of-the-money, contributing nothing to our estimated option value. In other words, much of our computational effort will be wasted. Alternatively, we

can change the drift in our simulations, so that more of them will finish in-the-money. Provided we appropriately weight the result using a likelihood ratio, we will get the correct option value. In particular, let $S_T^{i,\mu}$ be the terminal stock price using a different drift rate μ . Then

$$\widehat{V}_{\text{IS}} = \frac{1}{M} \sum_{i=1}^M V(S_T^{i,\mu}) e^{-rT} L_i, \quad (31)$$

where L_i is the likelihood ratio. In the case of geometric Brownian motion,

$$L_i = \left(\frac{S_T^{i,\mu}}{S_0} \right)^{(r-\mu)/\sigma} \exp\left(\frac{(\mu^2 - r^2)T}{2\sigma^2} \right) \times \exp\left(\frac{(r - \mu)T}{2} \right), \quad (32)$$

where S_0 is the current stock price. (We note that the expression given on p. 1284 of [9] is not correct as it omits the second term involving the exponential function.) Note that we can also change the volatility, or even the distribution function itself, provided that the new distribution has the same support as the old one.

There are several other possibilities, including *moment matching*, *stratified sampling*, and *low-discrepancy sequences*. In the latter, the ‘random’ numbers are not random at all, rather they are generated in a particular deterministic fashion, designed to ensure that the complete probability space is always covered in a roughly uniform fashion, no matter how many draws have been made. (Of course, any computer-based random number generator will not produce truly random numbers, but it is the uniformity property that distinguishes low discrepancy sequences.) Low discrepancy sequences can be

particularly appealing in certain cases because they offer the potential for convergence at a rate proportional to M^{-1} , rather than $M^{-1/2}$ as for standard methods. Detailed descriptions of low discrepancy sequences and other techniques may be found in sources such as [4, 9, 36, 38].

Table 3 contains some illustrative results for our European put example using basic Monte Carlo, antithetic variates, a control variate, and importance sampling. The control variate used was the actual stock price itself, which has a present discounted value equal to the current stock price. We estimate β using an independent set of $M/10$ runs in each case. The calculated values of β were close to -0.25 for each value of M . In the case of importance sampling, we simply change to another geometric Brownian motion with a new drift of $\mu = -0.10$, keeping the same volatility. Estimated values and standard errors are shown for values of M ranging from one thousand, by factors of 10, up to one million. It is easy to observe the convergence at a rate of $M^{-1/2}$ in Table 3, as the standard errors are reduced by a factor of around three each time M is increased tenfold. On this simple problem, it seems that antithetic variates works about as well as importance sampling, with the control variate technique doing relatively poorly (of course, this could be improved with a better choice of control variate).

The main virtues of Monte Carlo methods are their intuitive appeal and ease of implementation. However, for simple problems such as the one considered here, they are definitely inferior to alternatives such as the binomial method or more sophisticated numerical PDE techniques. This is because of their relatively slow convergence rate. However, as will be noted below, there are situations in which Monte Carlo methods are the only methods that can be used.

Table 3 Illustrative results for Monte Carlo methods for a European put option. The parameters are $S = K = 100$, $r = 0.08$, $T = 1$ year, and $\sigma = 0.30$. The exact value is 8.0229. The control variate technique used the underlying stock price as a control. The parameter β in (29) was estimated using a separate set of simulations, each with 1/10th of the number used for the calculations reported here. The importance sampling method was used with a drift of $\mu = -0.10$ and a volatility of $\sigma = 0.30$

No. of simulations	Basic Monte Carlo		Antithetic variates		Control variates		Importance sampling	
	Value	Std. error	Value	Std. error	Value	Std. error	Value	Std. error
1000	7.4776	0.3580	7.7270	0.1925	7.5617	0.2514	8.0417	0.2010
10 000	8.0019	0.1187	7.9536	0.0623	7.9549	0.0809	8.0700	0.0635
1 00 000	8.0069	0.0377	8.0243	0.0198	8.0198	0.0257	8.0107	0.0202
1 000 000	8.0198	0.0120	8.0232	0.0063	8.0227	0.0082	8.0190	0.0064

Two other observations are worthy of note. First, in many cases it is possible to achieve enhanced variance reduction by using combinations of the techniques described above. Second, we have not described how the random numbers are generated. Details about this can be found in sources such as [36, 46].

Some Considerations in More Complex Situations

The discussion thus far has been intended to convey in some detail the types of methods generally available, but it has been almost entirely confined to the simple case of standard European style equity options on a single underlying asset. We now examine some of the factors that affect the implementation of the methods in a variety of more complicated settings.

We begin by considering *American options*. We briefly described a procedure for valuing these earlier in the case of the binomial method. Obviously, the same procedure could be used for any type of numerical PDE method; given our knowledge of the solution at the next time level (since we are solving backwards in time), we know the value of the option if it is not exercised. We can compare this with its value if it were to be exercised and take the higher of these two possibilities at each node. While this procedure works fairly well in terms of calculating accurate prices in an efficient manner, it is not without flaws. These are easiest to describe in the numerical PDE context. The method sketched above involves advancing the solution from one time level to the next, and then applying the constraint that the solution cannot fall below its value if the option were immediately exercised. An alternative is to solve the linear equation system subject to the constraint rather than applying the constraint afterwards. This imposes some additional computational overhead, since we are now solving a (mildly) nonlinear problem. Various techniques have been proposed for doing this (see [20, 22, 25], and references therein). This kind of approach has the following advantage. Consider the ‘smooth-pasting’ condition, which says that the delta of an American option should be continuous. This does not hold at the early exercise boundary, if we do not solve the system subject to the early exercise constraint. Consequently, estimates of delta and gamma can be inaccurate near the exercise boundary, if we follow the simple procedure of applying the constraint after the system is

solved. While in the numerical PDE context, it is worth reemphasizing our earlier observation that it is not possible, in general, to attain convergence at a second-order rate, using an algorithm with constant timesteps [25]. Intuitively, this is because near the option’s expiry the exercise boundary moves very rapidly, so small timesteps are needed. Techniques for achieving this speed of convergence include using the automatic timestep size selector described above from [37], as proposed in [25], or a suitably modified version of the adaptive mesh scheme described in [24].

The use of Monte Carlo methods for American options is more complicated. This was long thought to be an impossible task. The basic reason is that we must simulate forward in time. As we do so, we know at any point in time and along any path what the immediate exercise value is, but we do not know what the value of holding onto the option is (observe the contrast with numerical PDE methods where we solve backwards in time, so we do know the value of keeping the option alive). In recent years, however, there has been considerable progress towards devising ways of handling early exercise constraints using simulation. This literature started with a path-bundling technique in [54]. While this first attempt was not very successful, it did provide an impetus for several other efforts [5, 15, 17, 41, 48, 50]. Most of these provide a lower bound for the true American option price due to an approximation of the optimal exercise policy. Exceptions include [15], in which algorithms are derived that compute both upper and lower bounds, and [50]. A comparison of several approaches can be found in [27].

Dividends need to be treated in one of two ways. If the underlying asset pays a continuous dividend yield q , then for a Monte Carlo method we simply need to change the drift from r to $r - q$. The analogue in the numerical PDE context is to change the rSV_S term in (3) to $(r - q)SV_S$. Of course, the same idea applies in various other contexts (e.g. q is the foreign risk free rate in the case of options on foreign exchange). If the dividend paid is a specified amount of cash at a discrete point in time, we need to specify the ex-dividend price of the stock. By far, the most common assumption is that the stock price falls by the full amount of the dividend. This is easy to handle in Monte Carlo simulation, where we are proceeding forward in time, but a little more complicated for a numerical PDE method.

The reason is that as time is going backwards, we need to enforce a no-arbitrage condition that says that the value of the option must be continuous across the dividend payment date [56]. If t^+ indicates the time right after the dividend D is paid, and t^- is the time right before, then $V(S, t^-) = V(S - D, t^+)$. This will usually require some kind of interpolation, as $S - D$ will not typically be one of our grid points.

Discontinuous payoffs such as digital options can pose some difficulty for Monte Carlo methods, if hedging parameters are calculated in certain ways [14] (convergence will be at a slower rate). On the other hand, it has been argued that such payoffs pose more significant problems for numerical PDE (and binomial tree) methods, in that the rate of convergence to the option price itself will be reduced [31]. Crank–Nicolson methods will also be prone to oscillations near discontinuities, unless very small timesteps are taken. However, there is a fairly simple cure [45]. Following Rannacher [47], if we (a) smooth the payoff function via an l_2 projection; and (b) take two implicit timesteps before switching to Crank–Nicolson, we can restore quadratic convergence. This does not guarantee that oscillations will not occur, but it does tend to preclude them. (For additional insurance against oscillations, we can take more than two implicit timesteps. As long as the number of these is fairly small, we will converge at a second-order rate.) We will refer

to this as the *Rannacher procedure*. To illustrate, consider the case of a European digital call option paying 1 in six months, if $S > K = 100$ with $r = 0.05$ and $\sigma = 0.25$. Table 4 shows results for various schemes. If we use Crank–Nicolson with a uniform S grid and constant timesteps, we get slow (linear) convergence. However, this is a situation in which it is advantageous to place a lot of grid nodes near the strike because the payoff changes discontinuously there. If we try this, though, we actually get very poor results. The reason, as shown in the left panel of Figure 3, is a spurious oscillation near the strike. Of course, the results for the derivatives of the value (delta and gamma) are even worse. If we keep the variable grid, and use the Rannacher procedure with constant timesteps, Table 4 shows excellent results. It also demonstrates that the use of an automatic timestep size selector provides significant efficiency gains, as we can achieve basically the same accuracy with about half as many timesteps. The right panel of Figure 3 shows the absence of oscillations in this particular case when this technique is used. It is also worth noting how costly an explicit method would be. Even for the coarsest spatial grid with 121 nodes, the stability restriction means that more than 5000 timesteps would be needed.

Barrier options can be valued using a variety of techniques, depending on whether the barrier is presumed to be monitored continuously or discretely. The continuous case is relatively easy for a numerical

Table 4 Illustrative results for numerical PDE methods for a European digital call option. The parameters are $S = K = 100$, $r = 0.05$, $T = 0.5$ years, $\sigma = 0.25$. The exact value is 0.508 280

No. of grid nodes	No. of timesteps	Value	No. of grid nodes	No. of timesteps	Value
Crank–Nicolson, uniform S grid, constant timestep size			Crank–Nicolson, nonuniform S grid, constant timestep size		
101	50	0.564 307	121	50	0.589 533
201	100	0.535 946	241	100	0.588 266
401	200	0.522 056	481	200	0.587 576
801	400	0.515 157	961	400	0.587 252
1601	800	0.511 716	1921	800	0.587 091
Rannacher procedure, nonuniform S grid, constant timestep size			Rannacher procedure, nonuniform S grid, automatic timestep size selector		
121	50	0.508 310	121	26	0.508 318
241	100	0.508 288	241	48	0.508 290
481	200	0.508 282	481	90	0.508 283
961	400	0.508 281	961	176	0.508 281
1921	800	0.508 280	1921	347	0.508 280

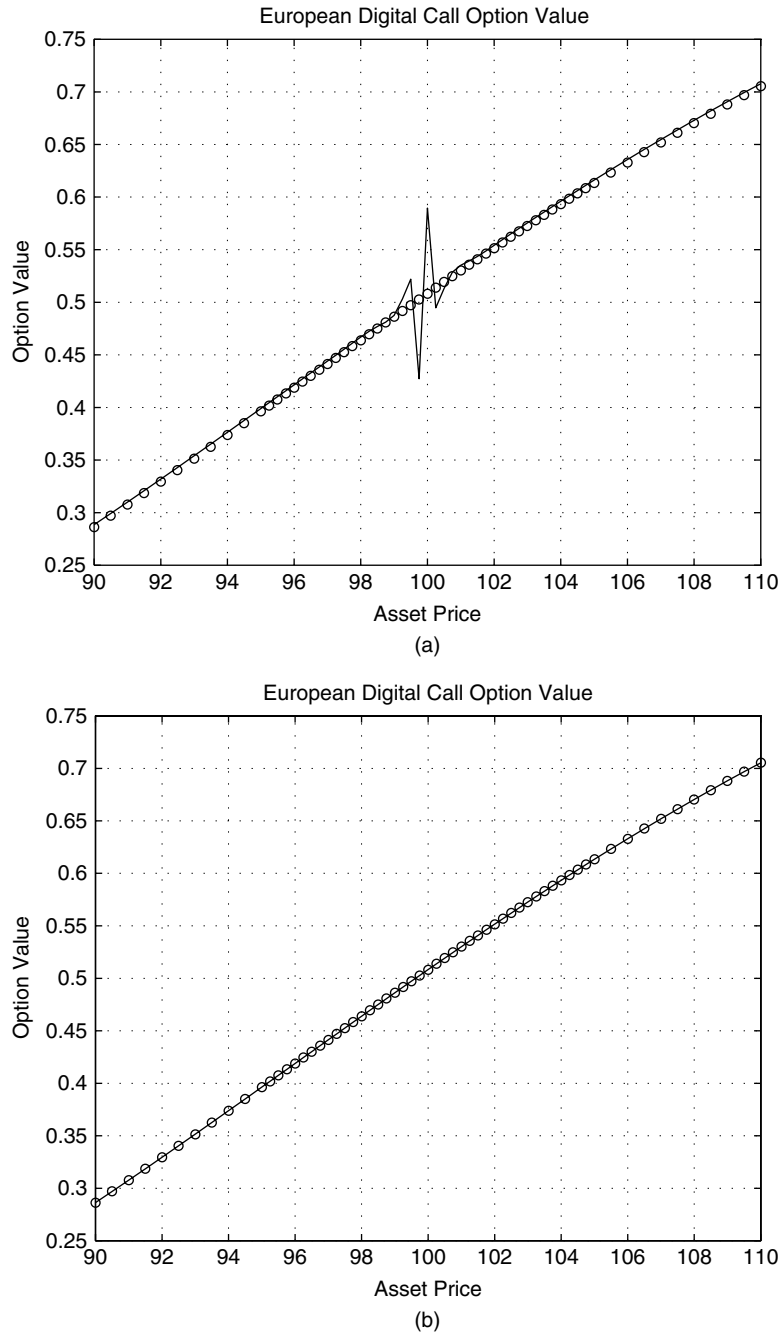


Figure 3 European digital call option value. Left: Crank–Nicolson using a nonuniform spatial grid with constant timesteps; Rannacher procedure using a nonuniform spatial grid with automatic timestep size selector. The parameters are $K = 100$, $r = 0.05$, $T = 0.5$ years, $\sigma = 0.25$. Circles depict the analytic solution, the numerical solution is shown by the solid lines

PDE approach, since we just have to apply boundary conditions at the barrier. However, it is a little trickier for a Monte Carlo approach. This is because, as we simulate forward in time using (25), a systematic bias is introduced, as we do not know what happens during the time intervals between sampling dates (that is, a barrier could be breached within a timestep). This implies that we overestimate prices for knock-out options and underestimate prices of knock-in options. Techniques for correcting this are discussed in [3, 16]. This is more of an academic issue than a matter of practical concern, though, as the overwhelming majority of barrier options feature discrete monitoring. In this case, Monte Carlo simulations are unbiased (provided that sampling dates for our simulations coincide with barrier observation times) but slow to converge. There are several papers dealing with adaptations of binomial and trinomial trees to handle barrier options [11, 12, 19, 24, 49]. Basically, all of these papers struggle with the issue that a fine grid is required near a barrier, and this then requires an enormous number of timesteps due to the tight coupling of grid spacing and timestep size in an explicit type of method. Of course, implicit or Crank–Nicolson methods do not suffer from this and can easily be used instead [62]. Note that we have to apply the Rannacher procedure described above at each barrier observation date if we want to have quadratic convergence because a discontinuity is introduced into the solution at every monitoring date.

Path-dependent options are an application for which Monte Carlo methods are particularly well suited. This is because we know the history of any path as we simulate it forward in time. Standard examples include Asian options (which depend on the average value of the underlying asset over time), look-back options (which depend on the highest or lowest value reached by the underlying asset during a period of time), and Parisian-style options (which depend on the length of time that the underlying asset is within a specified range). We simply need to track the relevant function of the stock price history through the simulation and record it at the terminal payoff date. One minor caveat is that the same issue of discretization bias in the case of continuously monitored barrier options also occurs here if the contract is continuously monitored (again, this is not of paramount practical relevance as almost all contracts are discretely monitored). Also, note that

Asian options offer a very nice example of the effectiveness of the control variate technique. As shown in [39], using an option with a payoff based on the geometric average (which has an easily computed analytic solution) as a control, sharply reduces the variance of Monte Carlo estimates of options with payoffs based on the arithmetic average (which does not have a readily calculable analytic formula).

However, Monte Carlo methods are relatively slow to converge, and, as noted above, not easily adapted to American-style options. On the other hand, numerical PDE and tree-based methods are somewhat problematic for path-dependent options. This is because as we solve backwards in time, we have no knowledge of the prior history of the underlying stock price. A way around this is to use a state augmentation approach. A description of the general idea can be found in [56]. The basic idea can be sketched as follows, for the discretely monitored case. (The continuously monitored case is rare, and results in a two dimensional PDE problem with no second-order term in one of the dimensions [56]. Such problems are quite difficult to solve numerically. See [58] for an example in the case of continuously observed Asian options.) Consider the example of a Parisian knock-out option, which is like a standard option but with the proviso that the payoff is zero if the underlying asset price S is above a specified barrier for a number of consecutive monitoring dates. In addition to S , we need to define a counter variable representing the number of dates for which S has been over the barrier. Since this extra variable can only change on a monitoring date, in between monitoring dates we simply solve the usual PDE (3). In fact, we have to solve a set of problems of that type, one for each value of the auxiliary variable. At monitoring dates, no-arbitrage conditions are applied to update the set of pricing equations. Further details can be found in [55]. This idea of solving a set of one-dimensional problems can obviously rely on any particular one-dimensional solution technique. If binomial trees are being used, the entire procedure is often referred to as a ‘binomial forest’. Similar ideas can be applied to look-back or Asian options [35, 59]. In the case of Asian options, the auxiliary variable is the average value of the underlying asset. Because the set of possible values for this variable grows very fast over time, we generally have to use a representative set of possible values and interpolation. This needs to be done with

care, as an inappropriate interpolation scheme can result in an algorithm that can diverge or converge to an incorrect answer [26]. It should be noted that not all path-dependent pricing problems can be handled in this fashion. HJM [29] models of the term structure of interest rates, do not have a finite-dimensional PDE representation, except in some special cases. Neither do moving window Asian options (where the average is defined over a rolling period of, e.g. 30 days). The only available technique, in these cases, is Monte Carlo simulation.

Multidimensional derivative pricing problems are another area where Monte Carlo methods shine. This is because the computational complexity of PDE or tree type methods grows exponentially with the number of state variables, whereas Monte Carlo complexity grows linearly. The crossover point is usually about three: problems with three or fewer dimensions are better handled using PDE methods, higher dimensional problems are more suited to Monte Carlo methods (in fact, they are basically infeasible for PDE methods). There are no particular complications to using Monte Carlo in higher dimensions, except for having to handle correlated state variables appropriately [36]. Of course, reasons of computational efficiency often dictate that variance reduction techniques become very important. In this regard, low-discrepancy sequences are frequently the preferred alternative. For numerical PDE methods in general, pricing problems in more than one dimension are better handled using finite element or finite volume methods because they offer superior grid flexibility compared to finite differences [60]. Also, when using Crank–Nicolson or implicit methods, the set of equations to be solved is no longer tridiagonal, though it is sparse. This requires more sophisticated algorithms [46]. It should be observed that there have been several attempts to generalize binomial/trinomial type explicit methods to higher dimensions (e.g. [8, 10, 18, 34, 42]). The fundamental issue here is that it is quite difficult in general to extend (to higher than one dimension) the idea of having a scheme with positive coefficients that sum to one, and hence can be interpreted as probabilities. The source of the difficulty is correlation, and often what is proposed is some form of transformation to eliminate it. While such schemes can be devised, it should be noted that it would be very difficult to accomplish this for

pricing problems, where a particular type of spatial grid is required, such as barrier options. Moreover, the effort is in one sense unnecessary. While positivity of coefficients is a necessary and sufficient condition for the stability of an explicit method in one dimension, it is only a sufficient condition in higher than one dimension; there are stable explicit schemes in higher dimensions with negative coefficients.

Finally, although our discussion has been in the context of geometric Brownian motion, the same principles and ideas carry over to other contexts. One of the advantages of numerical methods (whether Monte Carlo or PDE) is their flexibility in handling different processes for the underlying state variables. It is fairly easy to adapt the methods considered here to other types of diffusion processes such as stochastic volatility models [30], implied volatility surfaces [2], or constant elasticity of variance models [13]. More difficulties arise in the case of mixed jump-diffusion processes. There has not been a lot of work in this area. PDE type methods for American options have been developed in [1, 57]. Simulation techniques for pricing barrier options in this context are considered in [44].

Conclusions and Recommended Further Reading

The field of pricing derivative instruments using numerical methods is a large and confusing one, in part because the pricing problems vary widely, ranging from simple one factor European options through American options, barrier contracts, path-dependent options, interest rate derivatives, foreign exchange derivatives, contracts with discontinuous payoff functions, multidimensional problems, cases with different assumptions about the evolution of the underlying state variables, and all kinds of combinations of these items. The suitability of any particular numerical scheme depends largely on the context. Monte Carlo methods are the only practical alternative for problems with more than three dimensions, but are relatively slow in other cases. Moreover, despite significant progress, early exercise remains problematic for simulation methods. Numerical PDE methods are fairly robust but cannot be used in high dimensions. They can also require sophisticated algorithms for solving large sparse linear systems of equations and

are relatively difficult to code. Binomial and trinomial tree methods are just explicit type PDE methods that work very well on simple problems but can be quite hard to adapt to more complex situations because of the stability limitation tying the timestep size and grid spacing together. In addition to the references already cited here, recent books such as [51–53] are fine sources for additional information.

References

- [1] Andersen, L. & Andreasen, J. (2000). Jump-diffusion processes: volatility smile fitting and numerical methods for option pricing, *Review of Derivatives Research* **4**, 231–262.
- [2] Andersen, L.B.G. & Brotherton-Ratcliffe, R. (1998). The equity option volatility smile: an implicit finite difference approach, *Journal of Computational Finance* **1**, 3–37.
- [3] Andersen, L. & Brotherton-Ratcliffe, R. (1996). Exact Exotics, *Risk* **9**, 85–89.
- [4] Barraquand, J. (1995). Numerical valuation of high dimensional multivariate European securities, *Management Science* **41**, 1882–1891.
- [5] Barraquand, J. & Martineau, D. (1995). Numerical valuation of high dimensional multivariate American securities, *Journal of Financial and Quantitative Analysis* **30**, 383–405.
- [6] Black, F. & Scholes, M. (1973). The pricing of options and corporate liabilities, *Journal of Political Economy* **81**, 637–659.
- [7] Boyle, P.P. (1977). Options: a Monte Carlo approach, *Journal of Financial Economics* **4**, 323–338.
- [8] Boyle, P.P. (1988). A lattice framework for option pricing with two state variables, *Journal of Financial and Quantitative Analysis* **23**, 1–12.
- [9] Boyle, P., Broadie, M. & Glasserman, P. (1997). Monte Carlo methods for security pricing, *Journal of Economic Dynamics and Control* **21**, 1267–1321.
- [10] Boyle, P.P., Evnine, J. & Gibbs, S. (1989). Numerical evaluation of multivariate contingent claims, *Review of Financial Studies* **2**, 241–250.
- [11] Boyle, P.P. & Lau, S.H. (1994). Bumping up against the barrier with the binomial method, *Journal of Derivatives* **1**, 6–14.
- [12] Boyle, P.P. & Tian, Y. (1998). An explicit finite difference approach to the pricing of barrier options, *Applied Mathematical Finance* **5**, 17–43.
- [13] Boyle, P.P. & Tian, Y. (1999). Pricing lookback and barrier options under the CEV process, *Journal of Financial and Quantitative Analysis* **34**, 241–264.
- [14] Broadie, M. & Glasserman, P. (1996). Estimating security price derivatives using simulation, *Management Science* **42**, 269–285.
- [15] Broadie, M. & Glasserman, P. (1997). Pricing American-style securities using simulation, *Journal of Economic Dynamics and Control* **21**, 1323–1352.
- [16] Broadie, M. & Glasserman, P. (1997). A continuity correction for discrete barrier options, *Mathematical Finance* **7**, 325–348.
- [17] Broadie, M., Glasserman, P. & Jain, G. (1997). Enhanced Monte Carlo estimates for American option prices, *Journal of Derivatives* **5**, 25–44.
- [18] Chen, R.-R., Chung, S.-L. & Yang, T.T. (2002). Option pricing in a multi-asset, complete market economy, *Journal of Financial and Quantitative Analysis* **37**, 649–666.
- [19] Cheuk, T.H.F. & Vorst, T.C.F. (1996). Complex barrier options, *Journal of Derivatives* **4**, 8–22.
- [20] Coleman, T.F., Li, Y. & Verma, A. (2002). A Newton method for American option pricing, *Journal of Computational Finance* **5**, 51–78.
- [21] Cox, J.C. & Ross, S.A. & Rubinstein, M. (1979). Option pricing: a simplified approach, *Journal of Financial Economics* **7**, 229–263.
- [22] Dempster, M.A.H. & Hutton, J.P. (1997). Fast numerical valuation of American, exotic and complex options, *Applied Mathematical Finance* **4**, 1–20.
- [23] Duffie, D. (2001). *Dynamic Asset Pricing Theory*, 3rd Edition, Princeton University Press, Princeton, NJ.
- [24] Figlewski, S. & Gao, B. (1999). The adaptive mesh model: a new approach to efficient option pricing, *Journal of Financial Economics* **53**, 313–351.
- [25] Forsyth, P.A. & Vetzal, K.R. (2002). Quadratic convergence of a penalty method for valuing American options, *SIAM Journal on Scientific Computation* **23**, 2096–2123.
- [26] Forsyth, P.A., Vetzal, K.R. & Zvan, R. (2002). Convergence of numerical methods for valuing path-dependent options using interpolation, *Review of Derivatives Research* **5**, 273–314.
- [27] Fu, M.C., Laprise, S.B., Madan, D.B., Su, Y. & Wu, R. (2001). Pricing American options: a comparison of Monte Carlo simulation approaches, *Journal of Computational Finance* **4**, 39–88.
- [28] Geman, H. & Yor, M. (1993). Bessel processes, Asian options, and perpetuities, *Mathematical Finance* **3**, 349–375.
- [29] Heath, D., Jarrow, R.A. & Morton, A.J. (1992). Bond pricing and the term structure of interest rates: a new methodology for contingent claims valuation, *Econometrica* **60**, 77–105.
- [30] Heston, S.L. (1988). A closed-form solution for options with stochastic volatility with applications to bond and currency options, *Review of Financial Studies* **6**, 327–343.
- [31] Heston, S. & Zhou, G. (2000). On the rate of convergence of discrete-time contingent claims, *Mathematical Finance* **10**, 53–75.
- [32] Heynen, P. & Kat, H. (1996). Discrete partial barrier options with a moving barrier, *Journal of Financial Engineering* **5**, 199–209.
- [33] Hull, J.C. (2002). *Options, Futures, & Other Derivatives*, 5th Edition, Prentice Hall, NJ.

- [34] Hull, J.C. & White, A. (1990). Valuing derivative securities using the explicit finite difference method, *Journal of Financial and Quantitative Analysis* **25**, 87–100.
- [35] Hull, J. & White, A. (1993). Efficient procedures for valuing European and American path-dependent options, *Journal of Derivatives* **1**, 21–31.
- [36] Jäckel, P. (2002). *Monte Carlo Methods in Finance*, John Wiley & Sons, UK.
- [37] Johnson, C. (1987). *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, Cambridge.
- [38] Joy, C., Boyle, P.P. & Tan, K.S. (1996). Quasi-Monte Carlo methods in numerical finance, *Management Science* **42**, 926–938.
- [39] Kemna, A.G.Z. & Vorst, A.C.F. (1990). A pricing method for options based on average asset values, *Journal of Banking and Finance* **14**, 113–129.
- [40] Kloeden, P.E. & Platen, E. (1992). *Numerical Solution of Stochastic Differential Equations*, Springer-Verlag, New York.
- [41] Longstaff, F.A. & Schwartz, E.S. (2001). Valuing American options by simulation: a simple least-squares approach, *Review of Financial Studies* **14**, 113–147.
- [42] McCarthy, L.A. & Webber, N.J. (2001). Pricing in three-factor models using icosahedral lattices, *Journal of Computational Finance* **5**, 1–36.
- [43] Merton, R.C. (1973). Theory of rational option pricing, *Bell Journal of Economics and Management Science* **4**, 141–183.
- [44] Metwally, S.A.K. & Atiya, A.F. (2002). Using Brownian bridge for fast simulation of jump-diffusion processes and barrier options, *Journal of Derivatives* **10**, 43–54.
- [45] Pooley, D.M., Forsyth, P.A. & Vetzal, K.R. (2003). Convergence remedies for nonsmooth payoffs in option pricing, *Journal of Computational Finance* **6**, 24–40.
- [46] Press, W.H., Teukolsky, S.A., Vetterling, W.T. & Flannery, B.P. (1993). *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Edition, Cambridge University Press, Cambridge.
- [47] Rannacher, R. (1984). Finite element solution of diffusion problems with irregular data, *Numerische Mathematik* **43**, 309–327.
- [48] Raymar, S. & Zwecher, M.J. (1997). Monte Carlo estimation of American call options on the maximum of several stocks, *Journal of Derivatives* **5**, 7–24.
- [49] Ritchken, P. (1995). On pricing barrier options, *Journal of Derivatives* **3**, 19–28.
- [50] Rogers, L.C.G. (2002). Monte Carlo valuation of American options, *Mathematical Finance* **12**, 271–286.
- [51] Seydel, R. (2002). *Tools for Computational Finance*, Springer-Verlag, New York.
- [52] Tavella, D. (2002). *Quantitative Methods in Derivatives Pricing: An Introduction to Computational Finance*, John Wiley & Sons, UK.
- [53] Tavella, D. & Randall, C. (2000). *Pricing Financial Instruments: The Finite Difference Method*, John Wiley & Sons, West Sussex, UK.
- [54] Tilley, J.A. (1993). Valuing American options in a path simulation model, *Transactions of the Society of Actuaries* **45**, 83–104.
- [55] Vetzal, K.R. & Forsyth, P.A. (1999). Discrete Parisian and delayed barrier options: a general numerical approach, *Advances in Futures and Options Research* **10**, 1–15.
- [56] Wilmott, P. (1998). *Derivatives: The Theory and Practice of Financial Engineering*, John Wiley & Sons, UK.
- [57] Zhang, X.L. (1997). Numerical analysis of American option pricing in a jump-diffusion model, *Mathematics of Operations Research* **22**, 668–690.
- [58] Zvan, R., Forsyth, P.A. & Vetzal, K.R. (1998). Robust numerical methods for PDE models of Asian options, *Journal of Computational Finance* **1**, 39–78.
- [59] Zvan, R., Forsyth, P.A. & Vetzal, K.R. (1999). Discrete Asian barrier options, *Journal of Computational Finance* **3**, 41–67.
- [60] Zvan, R., Forsyth, P.A. & Vetzal, K.R. (2001). A finite approach for contingent claims valuation, *IMA Journal of Numerical Analysis* **21**, 703–731.
- [61] Zvan, R., Vetzal, K.R. & Forsyth, P.A. (1998). Swing low, swing high, *Risk* **11**, 71–75.
- [62] Zvan, R., Vetzal, K.R. & Forsyth, P.A. (2000). PDE methods for pricing barrier options, *Journal of Economic Dynamics and Control* **24**, 1563–1590.

(See also **Complete Markets; Derivative Securities; Interest-rate Modeling; Stochastic Investment Models; Transaction Costs**)

K.R. VETZAL