ELSEVIER

Journal of Financial Economics 53 (1999) 313–351

JOURNAL OF Financial ECONOMICS

www.elsevier.com/locate/econbase

The adaptive mesh model: a new approach to efficient option pricing☆

Stephen Figlewski[a,*], Bin Gao[b]

[a]*Stern School of Business, New York University, New York, 44 West 4th Street, NY 10012, USA*
[b]*Graduate School of Business, University of North Carolina, Chapel Hill, NC 27599, USA*

Received 20 August 1997; received in revised form 13 August 1998; accepted 1 February 1999

Abstract

Most derivative securities must be priced by numerical techniques. These models contain "distribution error" and "nonlinearity error". The Adaptive Mesh Model (AMM) sharply reduces nonlinearity error by grafting one or more small sections of fine high-resolution lattice onto a tree with coarser time and price steps. Three different AMM structures are presented, one for pricing ordinary options, one for barrier options, and one for computing delta and gamma efficiently. The AMM approach can be adapted to a wide variety of contingent claims. For some common problems, accuracy increases by several orders of magnitude with no increase in execution time. © 1999 Elsevier Science S.A. All rights reserved.

JEL classification: G13; C63

Keywords: Adaptive Mesh; Option valuation; Lattice models; Barrier options; Numerical valuation techniques

* Corresponding author. Tel.: 212-998-0712.

E-mail address: sfiglews@stern.nyu.edu (S. Figlewski)

☆We would like to thank Jingzhi Huang, Nengjiu Ju, Matthew Richardson, Marti Subrahmanyam, the referee, seminar participants at Columbia University, the Fields Institute, Hong Kong University of Science and Technology, the University of Missouri, the University of North Carolina, the University of Oklahoma, Penn State University, Tulane University, Goldman Sachs, Morgan Stanley, and many others for their helpful comments and suggestions on earlier versions of this paper.

PII: S0304-405X(99)00024-0

1. Introduction

Closed-form valuation equations exist for only a small subset of all possible derivative securities. Most have to be priced by some numerical approximation technique, such as Binomial and Trinomial lattice models. These models are widely used because they are intuitive and very flexible. It can be proved that as the lattice is made finer, these methods converge to the theoretical option values that would be produced by a continuous-time, continuous-state model, such as Black–Scholes. Unfortunately, convergence is normally non-monotonic, and some problems require very large numbers of calculations to achieve acceptable accuracy. This typically occurs when the lattice is too coarse in some critical region where the option value is highly nonlinear. Examples include around the strike price at expiration or near the knock-out price for a barrier option. Reducing the step size increases accuracy, but the computational effort required increases very rapidly, with much of it wasted on unimportant regions.

In this paper, we present a new approach for constructing a lattice-based valuation model that allows the user to vary the resolution in different parts of the tree. A relatively coarse grid that is fast to calculate is used for most of the lattice, but a small section of fine mesh is constructed where greater accuracy really matters. This Adaptive Mesh Model (AMM) can improve efficiency significantly for a relatively small increase in computational effort. For some common problems, including calculation of delta, accuracy may be increased by several orders of magnitude relative to commonly used methods, in the same execution time.

The potential performance improvement is so large in some cases that entire classes of problems that could not reasonably be addressed with the standard technology will now become accessible. For example, computing an implied volatility (IV) requires solving an option valuation model repeatedly with different trial values for the volatility input. If the valuation algorithm takes 5 min to price some exotic option and it must be priced ten times, on average, to obtain an accurate IV, constructing an implied volatility data set of reasonable size becomes extremely time-consuming.[1] An AMM procedure that reduces computation time for the option to 5 s (an improvement that is quite possible for many problems) greatly extends the possibilities for research in this area.

In the next section, we discuss approximation error in lattice models and describe how it can be thought of as arising from two different sources. "Distribution error" occurs because, throughout the tree, the model attempts to approximate a continuous lognormal distribution with a discrete binomial or trinomial distribution. "Nonlinearity error" arises because the option value is nonlinear in the underlying asset price in a way that can not be captured

[1] This is especially true if numerical derivatives must be computed at each step in the optimization.

accurately by the discrete lattice. For an ordinary call or put this occurs around the strike price at expiration; other derivatives can undergo a discrete jump in value at a given asset price or date. In the Binomial model, nonlinearity error produces a peculiar even-odd convergence property, which causes the approximation error to alternate between two quite different values as the number of time steps in the tree goes from even to odd and back to even.

Section 3 illustrates the valuation of a put option with a simple Adaptive Mesh Model. This demonstrates both how the AMM procedure is implemented and how it can greatly reduce approximation error with very little increase in computation. Section 4 examines the problem of pricing a barrier option, in this case a down and out call. Here, significant nonlinearity error occurs not just at expiration but at every point in time when the asset price approaches the barrier. Moreover, because the valuation lattice must allow at least one price step between the initial asset price and the barrier, the number of nodes required for a standard trinomial tree explodes when the asset price starts close to the barrier. Section 5 discusses calculation of the "Greek letter" risk exposures in trinomial lattices. We first show that there is a large difference in accuracy among the standard approaches to computing delta and gamma, and then present a third type of AMM that improves accuracy further by adding a section of fine mesh around the initial node of the tree. The final section concludes with a brief discussion of further extensions to the AMM approach. In the Appendix, we provide a formal proof of convergence for one of our models, the barrier option AMM. This also serves as a model of how such proofs may be constructed for other AMM structures.

2. Valuing options with lattice models

The Black–Scholes (1973) (BS) model was the first rigorously justifiable closed-form equation for pricing European calls and puts based on observable parameters. Equally importantly, the no-arbitrage principle used to obtain the BS equation pointed the way toward theoretical valuation models for all types of contingent claims. Unfortunately, American options and other contracts with early exercise present a serious problem because, although the no-arbitrage principle still holds, usable closed-form valuation equations seldom exist.

The Black–Scholes methodology begins with the assumption that the underlying asset (which we will often refer to as the "stock") follows the logarithmic diffusion,

$$\mathrm{d}S/S = \mu\,\mathrm{d}t + \sigma\,\mathrm{d}z, \tag{1}$$

where $\mathrm{d}S$ denotes the change in the asset price S over the infinitesimal time interval $\mathrm{d}t$, μ and σ are the instantaneous mean and volatility, and $\mathrm{d}z$ represents standard Brownian motion.

With continuous trading and no transactions costs, an investor can follow a self-financing dynamic trading strategy to replicate a derivative security's future payoff exactly. Thus, to avoid profitable arbitrage the option value must equal the cost of the replicating portfolio. This leads to the fundamental partial differential equation (PDE) of contingent claims pricing. In certain cases, the PDE can be solved to give a closed form valuation equation; otherwise an approximate option value can be obtained using "finite difference" numerical solution techniques, as Brennan and Schwartz (1977) demonstrate in their examination of American put pricing. The explicit finite difference technique for solving the PDE is equivalent to a trinomial tree procedure, though many users find lattice methods more intuitive.[2] Still, the identity between the two implies that the AMM technique can also be used within a standard finite difference scheme.

The original Binomial model is based on the principle of option replication: Within the binomial tree, an option's payoff can be reproduced by trading a portfolio consisting of just the stock and the riskless asset. Other lattice schemes, including the Trinomial, do not admit option replication. However, under the standard assumptions of option pricing, it can be shown that the option's fair value is the same as it would be in a risk neutral world. In this case, the fair value can be obtained simply by calculating the expected payoff under the risk neutral distribution and discounting back to the initial date at the riskless interest rate. Whenever risk neutral valuation is possible, any approximation procedure based on a probability distribution that approximates the risk neutral distribution and converges to it in the limit can be used to price options correctly. We can therefore use a trinomial lattice, or a more complex structure, without losing the ability to compute unique option values.

2.1. Approximation

Lattice models provide powerful, intuitive and asymptotically exact approximations to the theoretical option values under Black–Scholes assumptions for American options and many other contingent claims that do not have closed form valuation equations. However, using a discrete-time discrete-state lattice for an asset whose price is actually generated by a logarithmic diffusion introduces approximation error in two related but distinct ways, which we refer to as distribution error and nonlinearity error.

Consider using a Trinomial model with N time steps to value a put option. At any step n, the true asset price distribution is a continuous lognormal density, but in the tree this is approximated by a finite set of trinomial probabilities. By

[2] See, for example, Hull and White (1990).

construction, the discrete and the continuous distributions have the same mean and variance overall, but the discrepancy between them still produces distribution error in the option value.

Along with the probabilities, discreteness also affects the node values. One can think of the value at a given node as representing the probability weighted average option price over a range of the continuous price space around the node. This is computed by multiplying the value of the option at that node's asset price by the node probability. But if the option value function is highly nonlinear, evaluating it at the single asset price for the node gives a poor approximation to the average value over the whole interval around that node. This produces nonlinearity error.

Fig. 1 illustrates the two sources of error graphically for a one year 100 strike put option as of its expiration date when the initial asset price $S_0 = 100$.[3] Suppose the trinomial lattice has been set up with nodes 0.5 points apart at this date. The figure shows four nodes corresponding to stock prices of 99, 99.5, 100, and 100.5. Viewed from date 0, the lognormal density in this region of the price space is given by the heavy dashed line, and the nodes are indicated by the four gray bars. Conceptually, the lattice breaks the continuous price space up into discrete intervals associated with the nodes in the lattice. The light dashed bars indicate how the true probability density is discretized over this price range. The payoff on the put is shown by the heavy solid line (with the scale given on the right y-axis). The contribution of a particular node to the time zero option value equals the discounted value of the node probability times the option payoff at the asset price for that node.

In the interval (99.25, 99.75] around the $S = 99.5$ node, the option payoff is linear, ranging from 0.75 at $S = 99.25$ to 0.25 at $S = 99.75$. The option value of 0.5 at $S = 99.5$ is the (unweighted) average payoff over the interval. However, the true probability density is not constant in the interval: it is a little lower than the lattice probability at the left end and a little above it at the right. In this case, the node probability is too high at the higher option payoffs and too low where the payoff is lower. This is distribution error, and it causes the lattice to overestimate the average option payoff in this interval.

Now consider the interval (99.75, 100.25] around the strike price of 100. Distribution error is still present, but now there is another considerably larger error due to the nonlinearity in the payoff function. The option value of zero at $S = 100$ obviously understates the average option payoff in the interval. Since the put payoff is linear at every stock price except 100, nonlinearity error at this date is entirely concentrated in the interval containing the strike price.

[3] The other parameters for the example are $r = 10\%$, and $\sigma = 0.25$.

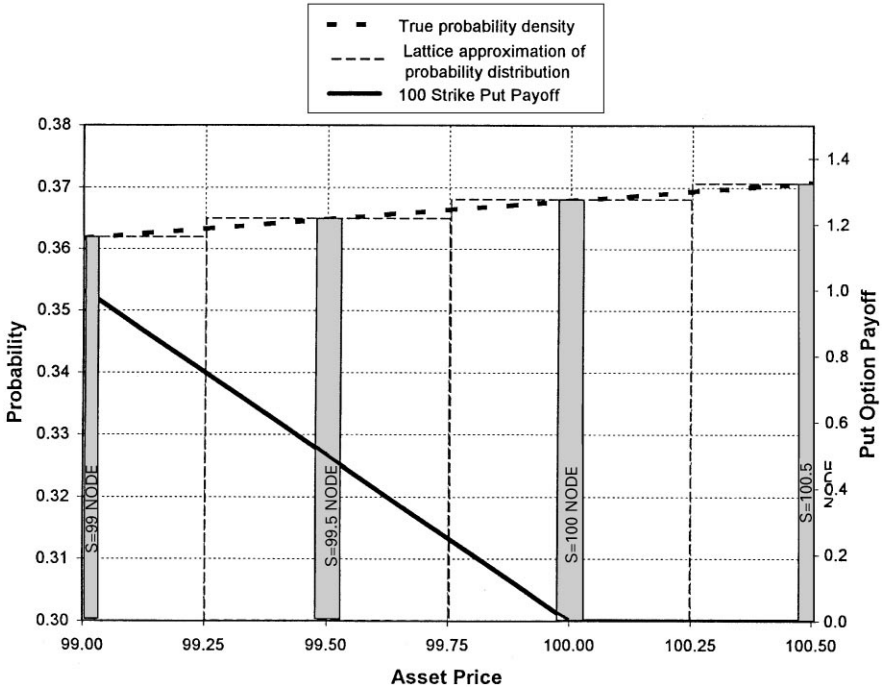

Fig. 1. Distribution error and nonlinearity error around the at the money nodes. The solid line represents the payoff on a one year, 100 strike European put option, with $r = 10\%$ and volatility 0.25. The gray shaded bars represent nodes in the trinomial lattice, corresponding to stock prices of 99.0, 99.5, 100.0, and 100.5. The heavy dashed line represents the lognormal density over this region of the price space. The light dashed bars indicate how the probability density is discretized over this price range. The contribution of a particular node to the time zero option value equals the discounted value of the node probability times the option payoff at the asset price for that node. The distribution error arises from the difference between the heavy dashed line and the light dashed line. At the $S = 100$ node, the option value of zero understates the probability weighted average payoff in this interval; this is the nonlinearity error.

In a binomial model, as one steps through the tree, the nodes at step $n + 1$ fall in the middle of the nodes in step n. For instance, both 99-step and 101-step trees in this case would have nodes at about the same prices. With nonlinearity error, this induces a peculiar "even–odd" property to convergence that can be surprisingly large. The trinomial also exhibits non-monotonic convergence, but not of such a striking form.

Note that we distinguish between distribution error and nonlinearity error because the Adaptive Mesh Model can minimize the latter over a given region of the tree. However, both errors stem fundamentally from the nonlinearity of the payoff function. For example, an instrument whose payoff is linear in the underlying asset price, like a forward contract, has neither distribution error nor

nonlinearity error in a properly designed lattice, as shown in Eq. (2):

$$C_{\text{APPROX}} = e^{-rT} E_{\text{APPROX}}[V(S_T)] = e^{-rT} V(E_{\text{APPROX}}[S_T])$$
$$= e^{-rT} V(E_{\text{TRUE}}[S_T]) = C_{\text{TRUE}}. \quad (2)$$

From the tree, the derivative instrument's value C_{APPROX} is the expected value, under the approximating distribution, of its payoff $V(S_T)$ discounted back to time zero. Since $V(.)$ is a linear function, the expected value operator can be taken inside $V(.)$. But by construction, the expected value of S_T is the same under the approximating distribution as under the true distribution, so the values for the derivative must be equal.

2.2. The Trinomial model

The binomial model is very intuitive but it has little flexibility to deal with more complex option problems. The Trinomial model has more degrees of freedom and has proven to be more useful and adaptable for many derivative applications. A trinomial tree is built to approximate the risk neutral distribution. Since the asset price is assumed to be lognormal, the tree is typically based on the log of S. Define $X^* \equiv \ln(S)$, which implies that X^* is normally distributed. Under risk neutrality, X^* follows the process

$$dX^*(t) = \alpha\, dt + \sigma\, dz,$$

where $\alpha = r - q - \sigma^2/2$, and q denotes the instantaneous rate of dividend payout.

In a trinomial tree, over the next time step the underlying asset price is allowed to move to one of three values, designated as up (u), down (d), and middle (m). Associated with these branches are three risk-neutral probabilities, p_u, p_d, and p_m. For many applications, the rate of convergence is enhanced if the tree is symmetrical. Therefore, for the first and third examples, we employ a change of variables and define $X(t)$, the variable whose evolution the tree is set up to approximate, by $X(t) \equiv X^*(t) - \alpha t$. $X(t)$ is the mean-adjusted value of the log of the asset price. In other words, $X(t)$ is the deviation of $\ln(S_t)$ from its expected value as of time 0.[4]

We set the middle node value to be no change and the up and down moves to be of equal magnitude. Let k denote the length of a time step and h be the size of an up or down move. Thus, over one time period X goes to $X + h$ with probability p_u, to $X - h$ with probability p_d, and remains unchanged with

[4] Note that depending on how the procedures we describe below are implemented, when we refer to an asset "price" in the tree, it may actually be the log of the dollar price, possibly with the mean subtracted. We assume it will be clear to the reader when and how the appropriate conversions need to be done.

probability p_m. (This uses two of the six available degrees of freedom in setting up the tree.)

The value of the time step k is determined by the option's maturity, T, and the number of time steps to be used for the tree, N:

$$k = T/N.$$

At each node, the system of three next period prices and three probabilities must obey three constraints. First, the tree approximates the risk neutralized asset price distribution, so for total expected return to equal the riskless interest rate, the expected log price change per period must be αk, making the expected change in $X(t)$ zero. Second, the standard deviation must be consistent with the known volatility of the underlying asset. Third, the probabilities must sum to one.

This leaves one degree of freedom. As Cho and Lee (1995) and Gao (1996) show, distribution error can be reduced if the trinomial tree is set up to match higher moments of the normal distribution, beyond the mean and variance. By choosing a symmetrical distribution, all odd-numbered moments of the trinomial will be zero, as they are for the normal. We can therefore employ the remaining degree of freedom to set the kurtosis in the tree equal to that of the normal.

The resulting system has four equations in four unknowns:

$$\begin{aligned}
&1 = p_u + p_m + p_d,\\
&E[X(t+k) - X(t)] = 0 = p_u h + p_m 0 + p_d(-h),\\
&E[(X(t+k) - X(t))^2] = \sigma^2 k = p_u h^2 + p_m 0 + p_d h^2,\\
&E[(X(t+k) - X(t))^4] = 3\sigma^4 k^2 = p_u h^4 + p_m 0 + p_d h^4.
\end{aligned} \tag{5}$$

Solving Eq. (5) for the probabilities yields:

$$p_u = 1/6, \quad p_m = 2/3, \quad p_d = 1/6, \quad h = \sigma\sqrt{3k}. \tag{6}$$

Fletcher (1991, Chapter 4) examines the analogous case of an explicit finite difference PDE solution technique when what we call nonlinearity error has been eliminated. He proves that in his framework, "high order convergence" is possible, such that the approximation error goes to zero faster than with any other set of probabilities.

As with the Binomial model, the option price is determined by starting at the known asset price contingent payoffs at maturity and rolling backward through the tree. Eq. (7) shows the calculation for a single node at date t and (mean-adjusted log) price X:

$$\begin{aligned}
C(X,t) = e^{-rk}(&p_u(h,k)C(X+h, t+k) + p_m(h,k)C(X, t+k)\\
&+ p_d(h,k)C(X-h, t+k)).
\end{aligned} \tag{7}$$

Note that for generality, Eq. (7) allows for the possibility that the probabilities may vary with h and k, even though in the current case of Eq. (6) they are fixed.

Nonlinearity error occurs in a lattice approximation when the true option value does not change proportionally as the asset price changes between two nodes. In this case, the option will have a large gamma (positive or negative) in that price range. For a European option, nonlinearity error is greatest around the strike price at expiration. It turns out that an American option's nonlinearity error is also largely accounted for by the error in the last time step, for the prices that bracket the strike price.[5]

The standard technique for reducing the approximation error in a lattice model is to use finer time steps. Unfortunately, this increases the number of nodes rapidly. Designating the initial node as $n = 0$, the Binomial model has $n + 1$ nodes at time step n, or $[(N + 1)^2 + (N + 1)]/2 = (N^2 + 3N + 2)/2$ nodes in total. A trinomial tree has $2n + 1$ nodes at each step, and $(N + 1)^2$ in total. If h is proportional to $\sqrt{k}$, to cut the size of the price step in half requires a tree with four times as many time steps and about 16 times as many nodes. Thus computation time increases sharply as greater accuracy is required.

The problem with increasing accuracy by using more time steps is that since finer resolution is often important only in one critical area, most of the additional computation may be largely wasted. What we would like in this case is a high resolution fine lattice in the region of the strike price at expiration, but one that is coarse and fast to calculate everywhere else. We call such a hybrid tree an Adaptive Mesh Model (AMM) and will now show how to construct one for the put valuation problem.

3. The adaptive mesh model for the American put

Fig. 2, showing the section of the pricing lattice in the immediate vicinity of the strike price in the last few periods before expiration, illustrates the Adaptive

[5] Prior to expiration, for the nodes around an American put's early exercise boundary, there will be an approximation error with regard to where exercise will occur. But this does not translate into significant error in valuing the option. The "smooth pasting" property of the American option value means that it is not highly nonlinear around the early exercise boundary.

One technique for reducing nonlinearity error in an American put lattice is to recognize that "early" exercise is impossible within the final time step before expiration. Thus the option is European at that point in time. Broadie and Detemple (1996) suggest eliminating the nonlinearity error at maturity by the simple expedient of substituting the Black–Scholes value for the option price at every node in the second to last time step, and then rolling back through the tree using the standard recursion shown in Eq. (7). While this does not take into account the possibility of exercising an actual option during that final interval, the improvement in accuracy of the approximation is very striking. Of course, this approach requires the existence of a closed form solution for the European option with one period to maturity.

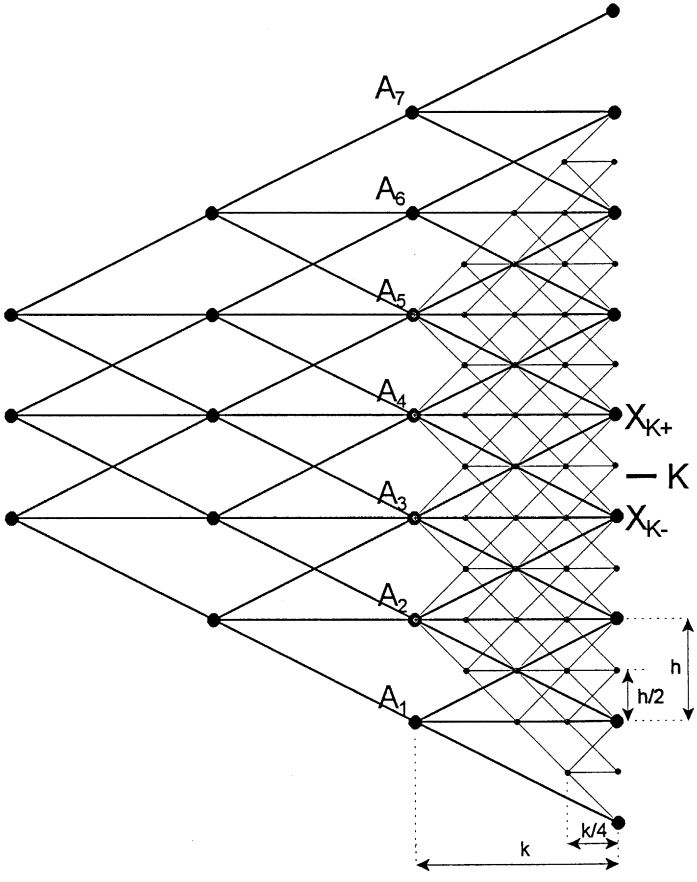

Fig. 2. An adaptive mesh model for the European put. This Trinomial tree shows the section of the pricing lattice in the immediate vicinity of the strike price in the last few periods before expiration. The coarse lattice, with price and time steps h and k, is represented by heavy lines. The fine mesh, with price and time steps $h/2$ and $k/4$, is represented by light lines. The fine mesh covers all $T-k$ coarse nodes from which there are both fine-mesh paths that end up in-the-money and fine-mesh paths that end up out of-the-money. K is the strike price, and X_{K-} and X_{K+} are the two date T coarse-mesh asset prices that bracket the strike price.

Mesh trinomial tree that we wish to construct to value a put option. The coarse lattice, with price and time steps h and k, is represented by the heavy lines. The objective is to build a fine-mesh tree around the strike price at expiration and join it to the coarse tree so that the valuation information is transmitted properly. Setting the fine price steps to be $h/2$ and, therefore, the time steps to be $k/4$ guarantees that fine-mesh nodes will overlap coarse-mesh nodes one coarse step before expiration. The fine mesh is used to obtain the option values for these overlapping nodes, labeled A_2 through A_5, while option values for the other

coarse-mesh nodes at this time step are computed in the usual way from the terminal payoffs in the coarse lattice. This incorporates the more accurate fine-mesh values for the critical nodes into the coarse tree. Valuation then proceeds by rolling back through the coarse lattice to the initial date.

As is clear from Fig. 2, the fine mesh is only added for dates between $T-k$ and T, in the region around the strike price. To be precise, it must cover all coarse nodes at date $T-k$ from which there are both fine-mesh paths that end up in-the-money and fine-mesh paths that end up out-of-the-money. For example, there is no need to use the fine mesh to obtain a value for node A_6 at $T-k$ because every possible fine-mesh path that would begin at that point would end up out-of-the-money, at a node where the option payoff is linear. However, from node A_5, while the coarse-lattice paths all end up out-of-the-money, a fine-mesh path with only down moves between $T-k$ and T would end up below K and in-the-money. Since it is around K that the option payoff is nonlinear, the fine-mesh value for node A_5 will be more accurate than what the coarse mesh would have produced.

Let us denote the two coarse-mesh date T asset prices that bracket the strike price as X_{K-} and X_{K+}. As Fig. 2 shows, the fine mesh starting at date $T-k$ coarse-mesh nodes must cover all of the time T prices in the range from $X_{K+}-2h$ to $X_{K-}+2h$. This increases the number of nodes to be calculated in this area from 12 in the coarse-mesh lattice to 52 in the fine mesh (4 at date $T-k$, plus 9, 11, 13, and 15 for dates $T-3/4k, \ldots, T$). Thus, even for a trinomial tree with as few as 20 steps, introducing an adaptive mesh increases the computational burden less than just adding one time step. In a 100-step trinomial tree, adding one level of adaptive mesh increases the total number of node calculations by less than 0.4%.

An important feature of the AMM approach is that it is isomorphic at successive levels of refinement. That is, once an AMM tree is constructed as in Fig. 2, it is simple to add another level of still finer mesh, with price steps of size $h/4$, by following exactly the same procedure as we have just described for the period $T-k/4$ to T. The total "cost" of this further refinement is just 40 additional node calculations.

Table 1 presents comparative performance statistics for the standard Binomial and Trinomial models and the AMM with one and two levels of fine mesh added around the strike price at expiration. The models are used to compute theoretical values, deltas, and gammas for a test set of 27 European put options.[6] European options are used to enable us to compare the accuracy of the various approximations against an exact benchmark. Obviously, there would be no need to use any of these methods to price European options in practice, but

[6] This is a standard test set that has been examined by Geske and Johnson (1984), among others.

Table 1
Performance of binomial, trinomial, and adaptive mesh models in valuing European puts

This table compares the performance of different lattice-based approximation methods in computing the theoretical price, delta and gamma for a set of 27 European put options. Root mean squared errors (RMSE) relative to the exact Black–Scholes values and computation times averaged over 10 identical runs for each set of parameters on a Dell Pentium Pro 200 MHz PC are displayed for the Cox, Ross, and Rubinstein (1979) version of the Binomial model, the standard Trinomial model, and the Adaptive Mesh Model with 1 and 2 levels of fine mesh around the strike price at maturity (denoted AMM-1 and AMM-2, respectively). The models are based on an initial stock price of 40, a riskless interest rate of 5.0% (4.88% continuously compounded), and no dividends. The option test set includes European puts with all 27 combinations of: strike prices of 35, 40, and 45; maturities of 1, 4, and 7 months; and volatilities of 0.20, 0.30, and 0.40.

Model	Time steps	Approximation RMSE			Nodes	Execution time (s)
		Price	Delta	Gamma		
Binomial	25	0.020841	0.005805	0.001594	351	0.0060
Trinomial	25	0.012025	0.003337	0.000428	676	0.0090
AMM-1	25	0.002812	0.003345	0.000539	716	0.0117
AMM-2	25	0.000615	0.003359	0.000548	756	0.0121
Binomial	100	0.004929	0.001470	0.000197	5151	0.0451
Trinomial	100	0.002770	0.000846	0.000144	10201	0.0941
AMM-1	100	0.000600	0.000845	0.000140	10241	0.0961
AMM-2	100	0.000151	0.000854	0.000138	10281	0.0982
Binomial	250	0.002214	0.000534	0.000073	31626	0.2163
Trinomial	250	0.001360	0.000346	0.000061	63001	0.5407
AMM-1	250	0.000245	0.000334	0.000056	63041	0.5418
AMM-2	250	0.000057	0.000342	0.000056	63081	0.5428
Binomial	1000	0.000448	0.000145	0.000020	501501	3.0674
Trinomial	1000	0.000244	0.000079	0.000015	1002001	8.5623
AMM-1	1000	0.000056	0.000086	0.000014	1002041	8.5954
AMM-2	1000	0.000016	0.000085	0.000014	1002081	8.5854

the efficiency gains shown in Table 1 are illustrative of what would be found for American puts (see footnote 5).

The models are based on an initial asset price of 40, riskless interest of 5% (corresponding to a 4.88% continuous rate), and no dividend payout. The 27 combinations include strike prices of 35, 40, and 45; maturities of 1, 4, and 7 months; and volatilities of 0.20, 0.30, and 0.40. The table shows the root mean squared error for the computed values relative to the exact Black–Scholes values, as well as the approximate execution time on a Pentium Pro 200 MHz computer. (There is a small amount of noise in the very short execution time estimates.) Lattices with 25, 100, 250, and 1000 time steps are examined.

The ability of the AMM approach to increase the accuracy of the price computation with virtually no increase in execution time is striking. The

AMM-2 model, for example, with only 25 time steps is much more accurate than a standard Trinomial with 250 time steps, and only a little less accurate than a 1000 step Binomial that requires about 250 times greater execution time. Comparing across the models for a given number of time steps, we see that computation times for the three trinomial-based models are about the same. The Binomial runs distinctly faster, but it is only about half as accurate as the standard Trinomial and much less accurate than the two AMM models. The AMM-1 model is about four times as accurate as the standard Trinomial. The AMM-2, with a second level of even finer mesh, is about four times as accurate as the AMM-1.[7]

The AMM approach has little effect on the delta and gamma calculations. Note that we do not compute delta and gamma as numerical derivatives by simply perturbing the starting asset price. As explained more fully below, that procedure is rather inaccurate and is greatly affected by the nonlinearity error problem. Instead, following Pelsser and Vorst (1994), we extend the tree back one period prior to the initial date and calculate the "Greek letters" from the node values within the extended tree. The computations then involve differencing option values at asset prices exactly one price step apart. Since nonlinearity error at expiration affects both option prices similarly, its effect is greatly reduced by the differencing. Section 5 examines the calculation of delta and gamma more carefully and shows how another AMM technique, this time applied to the initial nodes in the tree, can improve estimates of the Greek letter risk exposures.

4. An adaptive mesh model for barrier options

The previous section demonstrated that significant improvements in accuracy can be obtained with very little increase in computation time by adding a small section of fine resolution mesh in a single critical region of an option valuation lattice. This section shows how the AMM approach can be extended for use with barrier options and similar instruments whose values depend on whether the asset price reaches a specified level at any point during the option's life.

As the derivatives market has expanded, new instruments have been created with a wide variety of early exercise features and other more exotic contingencies.

[7] These results indicate a convergence rate both in the standard lattice models and as additional levels are added to the AMM that is approximately proportional to the finest time step (rather than to the price step, which only goes down with the square root of the time step). The effect of nonlinearity error on the option value is a function of the price step size times the probability of reaching the particular nodes where the error occurs, and that probability also falls with the square root of the time step. Thus, adding one level of fine mesh cuts the time step by a factor of four also reduces nonlinearity error by about a factor of four.

Closed-form solutions exist for some "plain vanilla" European instruments, but numerical approximation is frequently the only available valuation technique. Lattice-based methods for many such options can be enhanced enormously by an appropriate AMM procedure. This is particularly important for computationally intensive procedures like calculating implied volatilities.[8]

We apply the AMM technology to the common but difficult problem of pricing a barrier option when the initial asset price lies close to the barrier. A typical barrier option pays off at expiration like an ordinary call or put, except that the payoff is contingent upon whether the underlying asset price has reached the specified barrier price at some earlier point during the option's lifetime. For example, a "down-and-out" call option with strike price K and barrier price H (known as the "out-strike") has the same payoff at expiration as a European call, if the stock price stays above H throughout its entire life. But if the price ever falls to H or below, the option is "knocked-out" and expires worthless, regardless of the underlying asset price at expiration. Such a contract is often known as a "knock-out" option and the out-strike is the "knock-out barrier". Barrier options have become widespread, particularly for foreign currency contracts. There are also a variety of other instruments with similar kinds of contingent payoffs, including capped options, ladder options, and interest rate corridors.[9]

4.1. Valuing barrier options with a trinomial model

Nonlinearity error presents great difficulty for valuing these instruments with standard lattice-based techniques because price discreteness in the tree interacts with the price barrier. Convergence typically exhibits a choppy pattern similar to the even-odd behavior of the Binomial model for regular options. We will use the European down-and-out call option to illustrate the valuation problems of barrier options and the use of the AMM to deal with them. Notation is as above, with the addition of the symbol H to denote the option's knock-out barrier ($H < S$).

One advantage of this example is that there is a closed-form valuation equation that can serve as our benchmark for evaluating accuracy. Merton

[8] Calculating implied volatilities, even for ordinary American calls and puts, requires extensive computation to solve the valuation model repeatedly for each option. For example, when Canina and Figlewski (1993) used a 500 time step Binomial model to obtain implied volatilities for about 17,000 stock index calls, the required CPU time on a VAX mainframe computer was approximately one week. Obviously, computer speeds have increased since that time, but so has the complexity of derivative instruments.

[9] Rubinstein (1991) describes a number of exotic option contracts, many of which have barriers. Brief explanations of a vast array of traded derivative products, including many with various kinds of barrier features, are given in Gastineau and Kritzman (1996).

(1973) derives the following solution:

$$C_{\mathrm{DO}}(S, K, T, r, \sigma, H) = C_{\mathrm{BS}}(S, K, T, r, \sigma) - (H/S)^{2(r-(\sigma^2/2))} \times C_{\mathrm{BS}}(H^2/S, K, T, r, \sigma), \tag{8}$$

where C_{DO} denotes the down-and-out call value and C_{BS} is the Black–Scholes call formula.

The knock-out barrier is a fixed asset price over the option's lifetime. To have each layer of nodes fall at the same price in every time step, a trinomial tree for pricing barrier options is typically set up slightly differently from the one examined above. For example, to have the out-strike exactly 2 price steps below the initial asset price at every date, the tree must be built around the initial log stock price without adjusting for the mean. We therefore set $X(t) = \ln(S_t)$ for all t. In this tree, keeping the middle node as no change and the up and down moves equal to h, the probabilities must now be changed so that the mean and variance in the tree match the risk neutral distribution. It is no longer possible to match the kurtosis too.

The conditions expressed in Eq. (5) must be modified. The expected value over one step must now be equal to αk and the second moment must be $\alpha^2 k^2 + \sigma^2 k$. The equation for the fourth moment is dropped and value for h is set separately. This results in three equations in three unknowns, that can be solved to give the probabilities as:

$$\begin{aligned} p_{\mathrm{u}}(h, k) &= \tfrac{1}{2}(\sigma^2(k/h^2) + \alpha^2(k^2/h^2) + \alpha(k/h)), \\ p_{\mathrm{d}}(h, k) &= \tfrac{1}{2}(\sigma^2(k/h^2) + \alpha^2(k^2/h^2) - \alpha(k/h)), \\ p_{\mathrm{m}}(h, k) &= 1 - p_{\mathrm{u}}(h, k) - p_{\mathrm{d}}(h, k). \end{aligned} \tag{9}$$

The size of the price step, h, is the remaining free parameter in the tree. But once k is set, not every h value will produce nonnegative probabilities for all three nodes in Eq. (9). In general, both k and h will be far below 1, so

$$p_{\mathrm{m}} = 1 - p_{\mathrm{u}} - p_{\mathrm{d}} \approx 1 - \sigma^2 k/h^2. \tag{10}$$

To guarantee nonnegativity for the probabilities, h should be of the same order as $\sqrt{k}$. Define a free parameter $\lambda > 1$ such that

$$\lambda = h^2/\sigma^2 k. \tag{11}$$

A judicious choice for λ can improve the convergence properties of a tree substantially. We saw above that $\lambda = 3$ is a good choice for ordinary options, so we take that as the target value here as well.

Consider using the standard Trinomial model as given by Eqs. (7) and (9) to value a down-and-out call with the following parameter values:

$$S = 100, \quad K = 100, \quad T = 1 \text{ year}, \quad r = 10\%, \quad \sigma = 0.25, \quad H = 90.$$

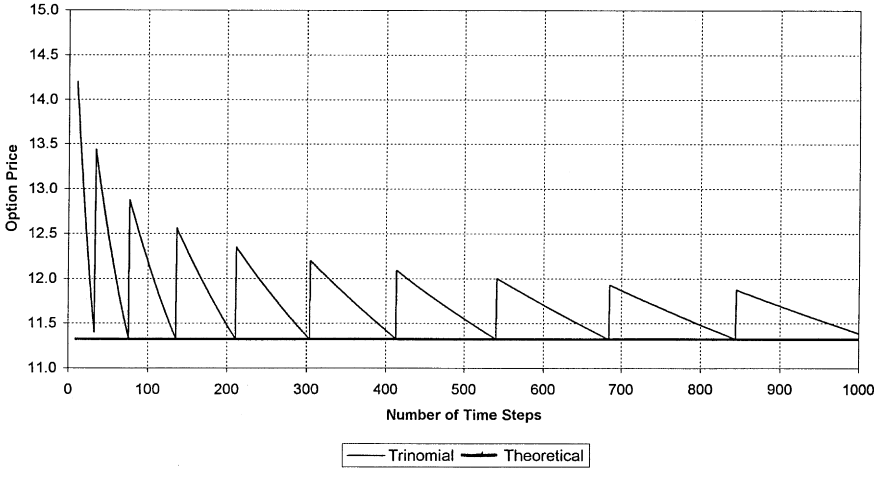

Fig. 3. Price of a down-and-out option using the standard trinomial method. A standard trinomial model is used to value a down-and-out call with the following parameter values: $S = 100$, $K = 100$, $T = 1$ yr, $r = 10\%$, volatility $= 0.25$, and $H = 90$. The horizontal line indicates the theoretical option value of 11.323, computed using Merton's (1973) analytical formula (Eq. (8)). The jagged line represents the trinomial model's approximation of the option value using different numbers of time steps.

Fig. 3 shows the convergence to this option's theoretical value of 11.323, as given by Eq. (8). As the number of time steps increases, the approximation approaches the correct price and then suddenly jumps away from it. Even with more than 1000 time steps (more than one million node calculations), the typical error is still unacceptably large.

The source of this problem is easily visible in Figs. 4 and 5. In Fig. 4, the barrier lies just slightly less than two price steps below the current asset price, so the option is knocked out if the price falls two steps below the initial price at any time prior to expiration. In Fig. 5, however, an increase in the number of time steps has reduced the price step just enough that two down moves still leaves the asset price a little short of the barrier. Three down moves will now be needed to knock out the option, which is a distinctly lower probability event. Thus, a small increase in the number of time steps has led to a very small change in the size of the price step, but within the lattice this produces a significant drop in the probability of the option being knocked out and therefore a sharp jump in its estimated value. Because the effective barrier (three price steps) lies further from the initial price than the true barrier, the down-and-out option will be overvalued by a lattice model. By the same token, an "in" option that pays off at maturity only if the barrier has been previously reached will generally be undervalued.

The ideal situation is for the barrier to coincide with a layer of nodes so that, within the framework of the discrete lattice, the approximate option value

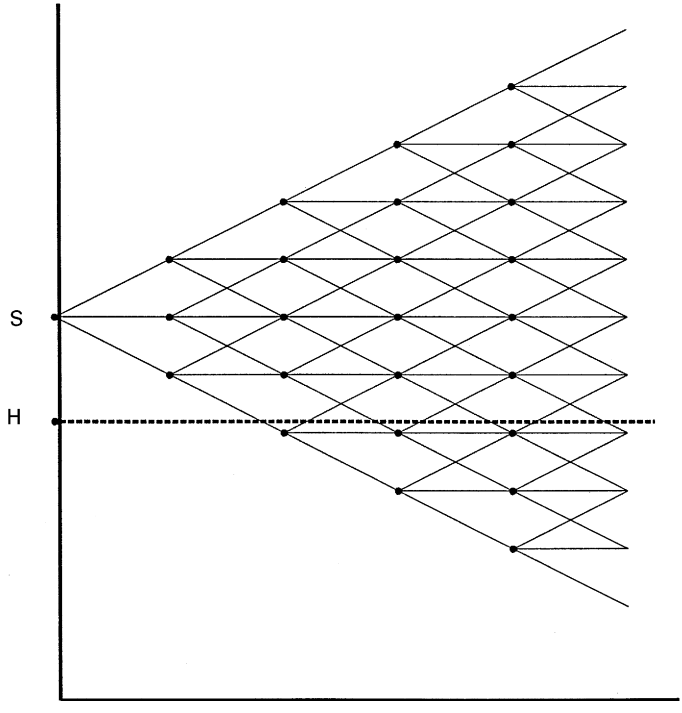

Fig. 4. A trinomial model for a barrier option. The barrier, H, lies just slightly less than two price steps below the current asset price, S. The option is knocked out if the price falls two steps below the initial price at any time prior to expiration.

incorporates an accurate estimate of the probability of hitting the barrier. This problem has been examined in the literature, and some creative solutions have been offered. One approach is to notice that since the size of the price step depends on the number of time steps, accurate valuation can be obtained simply by choosing the right value for N, as Fig. 3 illustrates. For example, Boyle and Lau (1994) show how to compute the sequence of optimal N values for a binomial tree. However, with only the limited freedom available in the Binomial, the method can not be extended to deal with greater complexity, such as multiple or time-varying barriers. Ritchken (1995) shows that for a trinomial barrier option model, performance can be enhanced significantly by restricting λ to be a value that produces an integer number of price steps between the current asset price and the barrier. We refer to that approach as the 'Restricted Trinomial Model' (RTM). With more degrees of freedom than the Binomial, the RTM can be fitted to match a second barrier also, but further generalization typically does not work.

These techniques face the greatest difficulty when the asset price is close to the barrier. For the barrier to coincide with a layer of nodes (and the contract not to

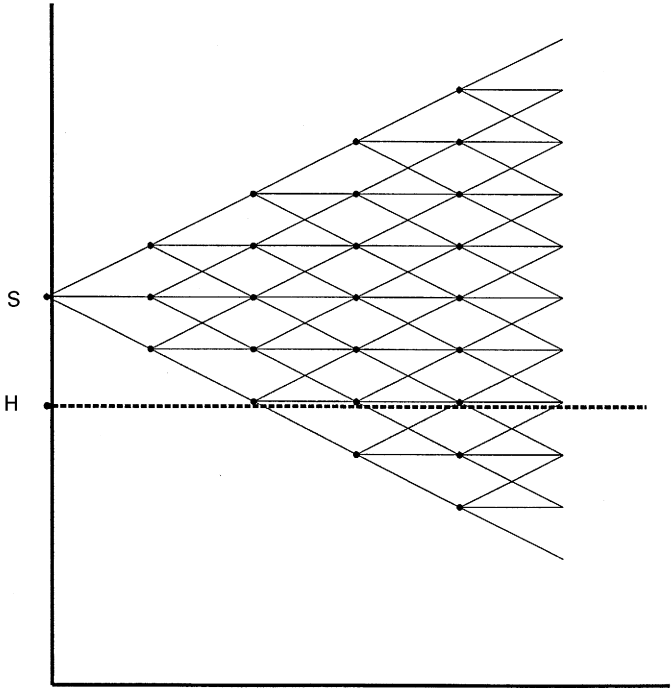

Fig. 5. A trinomial model with a shorter time step for a barrier option. This Trinomial model has one more time step than the model in Fig. 4. Consequently, the barrier, H, lies slightly *more* than two price steps below the current asset price S. The price must fall at least three steps below the initial price for the option to be knocked out. The probability that the price will fall three steps is less than the probability it will fall two steps. This model therefore produces a higher option value than the model in Fig. 4.

be already knocked-out) there must be at least one price step from the initial price to the barrier. But this imposes a maximum value for the price step h, equal to $\ln(S_0) - \ln(H)$. The price step, in turn, leads to a maximum time step, and therefore a minimum N that may be very large.

Consider valuing the down-and-out call described above with all of the same parameter values except that the current asset price is much closer to the barrier. For example, if $S_0 = 90.5$ the maximum price step that permits one down-move before hitting the barrier is $h = \ln(S_0) - \ln(H) = \ln(90.5) - \ln(90) = 0.00554$. Setting $\lambda = 3$ and plugging these values into Eq. (11) gives $k = 0.000164$.[10] The

[10] If there are 250 trading days to a year, and 7 trading hours in a day, a k value of 0.000164 years translates to a maximum time step of approximately 17.2 trading minutes. Alternatively, if every minute in the 365 day year were treated the same, whether the market is open or closed, this value of k produces a time step of 86.2 calendar minutes.

resulting tree therefore has $1/k = 6108$ time steps, which requires over 37 million node calculations. If one wanted a more accurate valuation, with two price steps to reach the barrier (which would be needed for an accurate estimate of gamma), the tree would have to have four times as many time steps (24,432) and almost 600 million nodes.

This leads to very slow convergence of the standard Trinomial for this problem. Even with 1000 time steps, the pricing error is unacceptably large: The estimated option value is 1.102 while the true value from Eq. (8) is 0.642, i.e., a 71.7% mispricing. The choppy convergence pattern seen in Fig. 3 does not produce nearly correct valuation for any relatively low N. The option value hits the minimum (most correct) value, and then jumps, when the number of price steps to reach the barrier changes. For this tree, the first jump does not occur until $N = 6109$, and the second is at $N = 24{,}436$.

4.2. Constructing an AMM lattice for a barrier option

An AMM can resolve this problem by constructing a section of high resolution lattice with the necessary fine price steps in the relevant region next to the barrier and joining it to a coarser lattice that allows efficient computation elsewhere. The flow of pricing information here is different than for the put option case, in which the fine mesh passed information "upward" to the coarse mesh. Here, the information flows from the coarse mesh downward to the fine mesh, allowing the barrier option to be valued properly at a stock price that is less than one coarse price step away from the barrier.

Fig. 6 shows the structure of the AMM tree that we wish to construct. The heavy lines indicate the coarse-mesh lattice, whose nodes are labeled A_{ij}, with i indicating the number of coarse-mesh price steps above the barrier and j the number of the coarse time step. Thus, A_{00} refers to the node falling on the barrier at date $t = 0$, and A_{10} is the price that is one coarse-mesh step above the barrier at date $t = 0$. The fine-mesh nodes are indicated by the letter B. We wish to compute the value of the down-and-out call at node B_{10}, which lies one half of a coarse-mesh price step above the knock-out price, i.e., at $\ln(H) + h/2$, at date $t = 0$.

Here is a brief overview of the process. The first task is to construct the coarse-mesh lattice and to compute option values at all of the A nodes. Next, the coarse mesh is used to compute option values at $\ln(H)$ and $\ln(H) + h$, for time intervals of length $k/4$ (following the dotted lines). These are the nodes that anchor the fine mesh to the coarse mesh. Finally, we fill in the remaining fine-mesh nodes for the price $\ln(H) + h/2$ at time steps of $k/4$ (the fine solid lines), culminating in the desired option value at node B_{10}.

We now describe these steps in more detail. Since we want B_{10} to correspond to the initial asset price, the value of h must be set to exactly $2(\ln(S_0) - \ln(H))$. This determines the asset price for the coarse-mesh lattice beginning at node

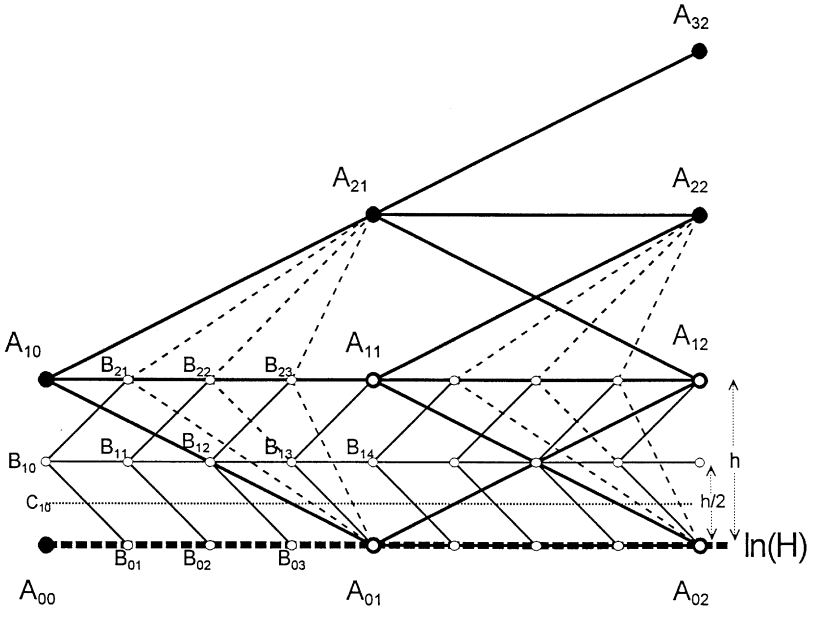

Fig. 6. Adaptive mesh model for a barrier option. The heavy lines indicate the coarse-mesh lattice, whose nodes are labelled A_{ij}, with i indicating the number of coarse-mesh price steps above the barrier and j the number of the coarse time step. The fine-mesh nodes are labelled B_{ij}. The barrier price is $\ln(H)$. To compute the option value at the initial asset price of B_{10}, first compute option values at all the A nodes. Second, use the coarse-mesh lattice to compute the option values at $\ln(H)$ and $\ln(H) + h$ for time intervals of $k/4$. Finally, calculate the remaining fine-mesh nodes for the price $\ln(H) + h/2$ at time intervals of $k/4$. The dotted lines indicate that nodes A_{01}, A_{11}, and A_{21}, are used to calculate option values at B_{21}, B_{22} and B_{23}. Similarly, the light solid lines indicate, for example, that the option value at node B_{10} is based on nodes B_{01}, B_{11}, and B_{21}. The fine dotted line indicates where the middle nodes of the next level of mesh would be placed to compute the option value at the initial asset price C_{10}.

A_{10}. Once the time step is set, as described below, the rest of the coarse lattice is filled out in the normal way.

The next step is to compute the option values for the points where the fine mesh is connected to the coarse mesh lattice. These B_{0j} and B_{2j} nodes lie along the barrier and at one price step h above the barrier, at time steps of length $k/4$. In this example, option values at the B_{0j} nodes are all zero, since they fall on the knock-out barrier.[11]

[11] Some barrier options pay a "rebate" R when they are knocked out, in which case the B_{0j} values are set to R. In cases with more complex behavior at the barrier, it may be necessary to solve for the B_{0j} lattice values, using the same procedure as we describe for the B_{2j} nodes in this tree.

The B_{2j} nodes that overlap A nodes are also known from the coarse lattice, e.g., the nodes that would be labeled B_{24} and B_{28} are the same as A_{11} and A_{12} (to limit clutter, they are not labeled separately in the figure). But the nodes like B_{23} that lie between coarse-mesh time steps require special handling. As Fig. 6 indicates, these are calculated from the nearest A nodes in the immediately subsequent time step. Thus B_{21}, B_{22}, and B_{23} are all obtained from A_{01}, A_{11} and A_{21}, while B_{25}, B_{26}, and B_{27} (not labeled) are derived from A_{02}, A_{12}, and A_{22}. This simply involves rolling backward using Eq. (7), with the only difference being that each of the B_{2j} nodes, falling on dates that are $k/4$, $2k/4$, and $3k/4$ before a coarse lattice time step, has its own set of up, middle, and down probabilities. These are obtained by replacing k in (9) by the appropriate effective time step size for that node, i.e., $3k/4$ for a B node like B_{21}, $2k/4$ for B_{22}, and $k/4$ for B_{23}.

For example, consider B_{23}. From Eq. (9), we have

$$p_u(h, k/4) = \frac{1}{2}\left(\sigma^2\frac{k/4}{h^2} + \alpha^2\frac{k^2/16}{h^2} + \alpha\frac{k/4}{h}\right),$$

$$p_d(h, k/4) = \frac{1}{2}\left(\sigma^2\frac{k/4}{h^2} + \alpha^2\frac{k^2/16}{h^2} - \alpha\frac{k/4}{h}\right),$$

$$p_m(h, k/4) = 1 - p_u(h, k/4) - p_d(h, k/4). \tag{12}$$

Plugging into Eq. (7) gives

$$C(B_{23}) = e^{-rk/4}(p_u(h, k/4)C(A_{21}) + p_m(h, k/4)C(A_{11}) + p_d(h, k/4)C(A_{01})), \tag{13}$$

where for convenience the notation has been adjusted to indicate explicitly the node to which the call value function $C(.)$ applies. Option values at the other B_{2j} nodes are computed in similar fashion using Eq. (9) with the appropriate multiple of $k/4$ for the time step.

Finally, when the B_{0j} and B_{2j} node values have been obtained for all j, we fill in the B_{1j} values by defining a third set of probabilities, for price steps of size $h/2$ and time steps of $k/4$. Since the dominant term in Eq. (9) is the one containing k/h^2, these probabilities will be almost the same as for the coarse mesh. The up, middle, and down probabilities used in calculating a B_{1j} node are given by

$$p_u(h/2, k/4) = \frac{1}{2}\left(\sigma^2\frac{k}{h^2} + \alpha^2\frac{k^2/4}{h^2} + \alpha\frac{k/2}{h}\right),$$

$$p_d(h/2, k/4) = \frac{1}{2}\left(\sigma^2\frac{k}{h^2} + \alpha^2\frac{k^2/4}{h^2} - \alpha\frac{k/2}{h}\right), \tag{14}$$

$$p_m(h/2, k/4) = 1 - p_u(h/2, k/4) - p_d(h/2, k/4).$$

To fill out these middle nodes, one must start from the expiration date and roll back through the tree with the normal recursion. This eventually gives the node B_{10} option value as

$$(B_{10}) = \mathrm{e}^{-rk/4}[p_{\mathrm{u}}(h/2, k/4)C(B_{21}) + p_{\mathrm{m}}(h/2, k/4)C(B_{11}) + p_{\mathrm{d}}(h/2, k/4)C(B_{01})]. \quad (15)$$

4.3. Setting the coarse mesh time step

Having shown how to construct the fine-mesh AMM lattice given the coarse-mesh tree, we now return to the coarse tree. It must be set up carefully to satisfy two conditions. First, as discussed above, the fine mesh must place the current asset price exactly one price step above the barrier, meaning

$$h = 2(\ln(S_0) - \ln(H)). \quad (16)$$

Second, the number of coarse-mesh time steps in the tree, $N = T/k$, must be an integer. Note that by setting the coarse time step correctly, all finer mesh sub-lattices will also have integer numbers of time steps.

The values of h and k have been linked through Eq. (11) to the parameter λ, which we set equal to 3 in the first example. For a barrier option, h and k must obey separate constraints, so the choice of λ is no longer free. Referring to it as the "stretch" parameter, Ritchken (1995) discusses how to choose λ to create a tree with a layer of price nodes that lies exactly on the out-barrier and an integral number of time steps.[12]

We need a variant of the procedure in this case, because the constraints are on the fine-mesh price step and the coarse-mesh time step. If λ is fixed at 3 (or at any other constant value), the price step from Eq. (16) normally does not produce an integral number of time periods in the option's life, so the tree must be stretched a little to fit properly. A simple procedure that changes λ only slightly is to calculate the noninteger number of steps the target λ would yield and then lengthen the time step just enough to eliminate the rounding error.

For example, setting $\lambda = 3$, $k_0 = h^2/3\sigma^2$. Let $N_0 = T/k_0$ be a noninteger number of length k_0 time steps in the option's lifetime. Eliminating the fractional step, the adjusted tree is built to have $N = \mathrm{int}(N_0)$ steps of length $k = T/N$. The resulting time step will be marginally longer than that produced by $\lambda = 3$, but the difference will be negligible for a typical lattice. (The step length increases by a fraction less than $1/N$, so $k - k_0 < k_0/N$.) The time step calculation can be expressed in a single general equation as

$$k = T/\mathrm{int}[(\lambda\sigma^2/h^2)T]. \quad (17)$$

[12] Ritchken actually defines λ slightly differently than we do: his λ is the square root of the λ given in Eq. (11).

4.4. The general AMM for a barrier option

This procedure shows how to construct an AMM lattice with one level of finer mesh at the knock-out barrier. Further refinement is easily introduced by using the identical approach to add additional levels of progressively finer mesh. The price step at each successive level will be half as large as at the previous one. The fine dotted line in Fig. 6 indicates where the middle nodes of the next level of mesh would be placed, in order to compute the option value at the initial asset price C_{10}.

The general AMM methodology for a barrier option is therefore the following. First choose M, the number of levels of fine mesh to be constructed. For the above example, $M = 1$. Computing the option value at C_{10} would involve a second level of fine mesh, making $M = 2$. The finest mesh price step is set to $\ln(S_0) - \ln(H)$, making the coarse mesh step h equal to

$$h = 2^M(\ln(S_0) - \ln(H)). \tag{18}$$

The coarsest mesh time step k is given by Eq. (17). First, the coarsest (A level) lattice is constructed starting at the initial (log) asset price $X = \ln(H) + h$. Then the finer levels (B, C, etc.) are added one at a time, following the procedure outlined above.

It is important that a numerical approximation be consistent, meaning that as the step size shrinks to zero, the option value from the model converges to the true continuous-time continuous-state theoretical value. The Appendix provides a proof of consistency for this AMM model. The method of proof is straightforward and can easily be applied to other AMM structures.

It is easy to see from Fig. 6 how many additional calculations are required to add a layer of fine mesh to a trinomial lattice. Consider the three layers in the B level. Along the upper and lower layers, for each coarse time step there are three new values to be computed for the B nodes that do not overlap coarse mesh nodes, e.g., B_{01}, B_{02}, B_{03}, B_{21}, B_{22}, and B_{23}. The middle B layer is entirely new, so there are four new nodes per coarse time step. The total for the first layer of finer mesh is therefore $3 + 3 + 4 = 10$ new nodes per time step, or 10N in total. Thus for a 100-step lattice that contains $101^2 = 10{,}201$ nodes, an AMM with a single level of fine mesh would add 1000 new nodes, making a total of 11,201. In contrast, increasing the resolution of a standard Trinomial model by halving the price step (and multiplying N by 4) would require increasing the total number of nodes to $401^2 = 160{,}801$.

Adding further levels of fine mesh requires ten new nodes for each time step of the next higher level. Thus the C level mesh in Fig. 6 requires 40 new nodes per coarse mesh time step and, in general, the mth level adds $10 \times 4^{m-1} \times N$ new nodes. Table 2 shows the number of nodes in an AMM lattice as finer levels of mesh are added and compares it to the number in a standard Trinomial model with the same mesh size everywhere. It is obvious that refining the standard

Table 2
Comparison of the number of nodes in a barrier option adaptive mesh model versus a standard trinomial model with the same sized price step as the finest level AMM mesh

This table compares the number of nodes in a barrier option AMM model with that in a standard Trinomial model. In each case, the Equivalent Trinomial model is configured to have the same sized price step as the finest AMM mesh. Differences are compared across models with 25, 100, and 400 time steps to maturity.

Model	$N = 25$	$N = 100$	$N = 400$
Standard trinomial	676	10,201	160,801
Adaptive mesh, $M = 1$	926	11,201	164,801
Equivalent trinomial	10,816	163,216	2,572,816
Adaptive mesh, $M = 2$	1926	15,201	180,801
Equivalent trinomial	173,056	2,611,456	41,165,056
Adaptive mesh, $M = 3$	5926	31,201	244,801
Equivalent trinomial	2,768,896	41,783,296	658,640,896
Adaptive mesh, $M = 4$	21,926	95,201	500,801
Equivalent trinomial	44,302,336	668,532,736	10,538,254,336

Trinomial to price barrier options near the boundary rapidly requires unmanageably large numbers of calculations. Of course, the higher resolution throughout the lattice produces smaller distribution approximation errors for the standard Trinomial than for the AMM, but there will be little difference between them in the nonlinearity error. That is largely determined by the size of the price step immediately adjacent to the barrier, which is the same for both models.

Table 3 shows how this difference translates into computation time for the RTM and the AMM. We value a one year down-and-out call with a strike price of 100 and an out-strike $H = 90$ for a sequence of initial stock prices that approach the barrier. In each case, the AMM uses a coarsest price step of about 2 points, but adds successive layers of fine mesh for asset prices closer to the barrier, while the RTM is constructed to have a price step equal to $\ln(S_0) - \ln(H)$.[13] (Computation times are for a personal computer with a 90 MHz Pentium processor and 32 megabytes of RAM.) The RTM computation became unmanageable (though not actually impossible) at an asset price of $90\frac{1}{8}$ because of the size of the lattice, while the AMM reached accurate answers almost instantaneously in every case. Note that the general rule that halving the price step increases the number of time steps by a factor of 4 and the

[13] Since the price step is defined in terms of the log of the price, the dollar values mentioned here are exact only at the barrier: it is h that is constant throughout the tree.

Table 3
Pricing a down-and-out call option with the AMM and the restricted trinomial model when the asset price is near the boundary

This table shows the analytic down-and-out call value from (8) and approximate values from the restricted trinomial model (RTM) and the adaptive mesh model (AMM). The RTM constructs a lattice with the same price step size everywhere and with a layer of nodes exactly on the knock-out boundary, one price step below the initial asset price. The AMM maintains the same size price step for the coarsest mesh but adds layers of finer mesh as the initial asset price approaches the barrier. M represents the number of layers of finer mesh. S_0 is the initial asset price. The parameter values are: $K = 100$, $r = 10\%$, $T - t = 1$ yr, $\sigma = 0.25$, and H (barrier) $= 90$. Computation time in seconds is for a PC with a Pentium 90 MHz processor and 32 megabytes of RAM. N/A indicates that the problem became unmanageable for the computer, as described in the text.

S_0	Analytic value	RTM			AMM		
		Value	Number of time steps	CPU time (s)	Value	AMM level	CPU time (s)
92	2.506	2.507	388	0.033	2.507	0	0.033
91	1.274	1.274	1535	0.750	1.274	1	0.050
$90\frac{1}{2}$	0.642	0.642	6108	12.35	0.643	2	0.059
$90\frac{1}{4}$	0.323	0.323	24,367	364.3	0.323	3	0.117
$90\frac{1}{8}$	0.162	N/A	97,335	N/A	0.162	4	0.317

total number of node calculations by a factor of 16 does not allow us to assume computation times will increase proportionally. As the size of the lattice increases, it becomes impossible to keep the whole tree in fast computer memory at one time and computation speed slows sharply. We therefore did not undertake the extensive reprogramming that would have been required to solve the RTM model starting at $S = 90\frac{1}{8}$.

5. An AMM for calculating delta and gamma

Academics tend to focus on derivative valuation, but in practice option pricing models are more important for risk management than for valuation. Traders adjust the volatility and other input parameters to make the model values match observed market prices. The most important risk exposures are measured by delta, the change in the option value for a small change in the asset price, and gamma, the change in delta. In a closed-form valuation model, these "Greek letter" risks can typically be obtained in closed-form also, as

$$\Delta = \partial C/\partial S,$$

$$\Gamma = \partial \Delta/\partial S = \partial^2 C/\partial S^2. \tag{19}$$

There are a variety of ways to estimate delta and gamma in a lattice model, some of which are distinctly better than others in terms of accuracy and speed of convergence. This section discusses the different standard approaches, all of which are subject to the effects of nonlinearity error, and then shows how accuracy can be enhanced by an AMM technique.

5.1. Delta and gamma in the trinomial model

In the binomial model, the number of shares in the replicating portfolio provides one estimate of delta. In contrast, the Trinomial is not based on option replication, and there is no comparable delta estimate. The most common technique for estimating delta and gamma in a trinomial model is to perturb the initial (log) price by some small amount, ε, and build new trees starting from $X_0 + \varepsilon$ and $X_0 - \varepsilon$. Unfortunately, nonlinearity error makes the estimates very noisy and, like the option price, they converge to the true values non-monotonically. Another major shortcoming is that the entire tree must be computed a second time to get delta and a third time to get gamma (or to get a delta centered on the initial asset price).

Since the perturbations are in log prices, the delta and gamma calculations must include the appropriate conversions, as follows:

$$\begin{aligned}
\Delta &= \frac{\partial C}{\partial S} = \frac{\partial C}{\partial \ln(S)}\frac{1}{S} \approx \frac{C(X_0+\varepsilon) - C(X_0-\varepsilon)}{2\varepsilon}\frac{1}{S},\\
\Gamma &= \frac{\partial \Delta}{\partial S} = \frac{\partial^2 C}{\partial S^2} = \frac{\partial}{\partial S}\left(\frac{\partial C}{\partial \ln(S)}\frac{1}{S}\right) = \left(\frac{\partial^2 C}{\partial (\ln(S))^2} - \frac{\partial C}{\partial \ln(S)}\right)\frac{1}{S^2}\\
&\approx \left(\frac{C(X_0+\varepsilon) + C(X_0-\varepsilon) - 2C(X_0)}{\varepsilon^2} - \frac{C(X_0+\varepsilon) - C(X_0-\varepsilon)}{2\varepsilon}\right)\frac{1}{S^2}.
\end{aligned} \tag{20}$$

Pelsser and Vorst (1994) discuss the problem of obtaining delta and gamma for a Binomial model and propose an alternative technique that increases efficiency markedly. Rather than starting from the initial asset price at time 0, they begin two time steps earlier and build the tree so that the middle node at the end of the tree's second period (time 0) falls at the desired starting asset price. This extended tree has three nodes at time 0, which permits calculating both delta and gamma centered on the right point in time. The method improves accuracy considerably, but its shortcoming is that the perturbation of the asset price is now one full price step in the lattice.

Fig. 7 illustrates economizing on extra calculations by extending the trinomial tree backwards in time. To obtain three asset prices for date 0, a Trinomial only needs to be extended backward for one period. The dashed lines indicate the

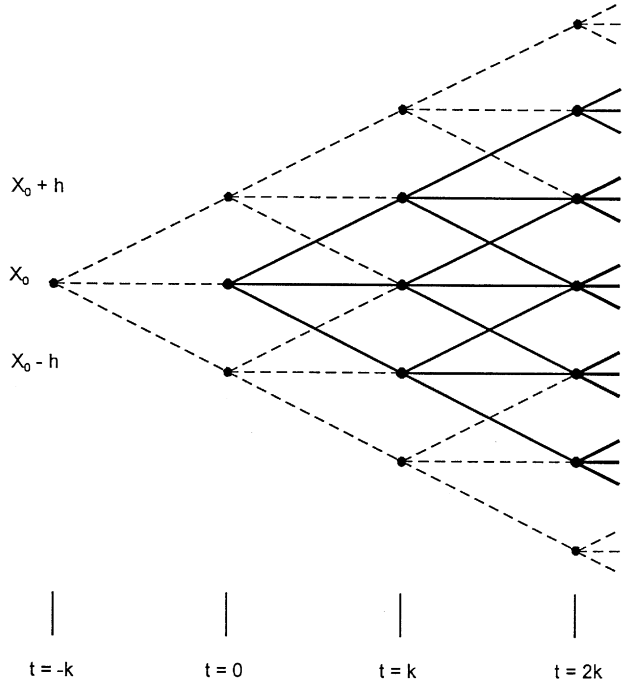

Fig. 7. Extending the tree to compute delta gamma. The trinomial tree is extended backward one period. The dashed lines indiate the new sections of lattice that must be added to the original tree. This produces three time zero option values at asset prices X_0, $X_0 + h$, and $X_0 - h$, from which delta and gamma can be calculated, without the necessity of constructing two new complete trees.

new sections of lattice that must be added to the original tree. It is apparent why extending the tree can be much more efficient than simply perturbing the initial asset price. Two-sided perturbations requires two new trees to be constructed starting from the perturbed time zero asset prices, but in Fig. 7, these trees overlap the existing lattice almost entirely. Only two new nodes are required at each time step from 0 to T, a total of $2N + 2$. (Since our real interest is in time 0, there is no need actually to compute a time $t = -k$ option price.) Delta and gamma values are obtained from Eq. (20) with the price step h substituted for ε.

5.2. *Relative performance of the standard methods in estimating Greek letter risks*

Table 4 compares these alternative approaches for estimating delta and gamma in the standard trinomial model. The table is set up like Table 1 and uses the same test set of 27 European put options. We report root mean squared errors relative to the exact values from the Black–Scholes

Table 4
Performance of the trinomial model in estimating delta and gamma for European puts

The table compares the performance of different approaches for estimating delta and gamma in the standard Trinomial model. "Perturbation" involves constructing new trinomial trees beginning from time 0 asset prices that are above and below the original (log) price X_0 by an amount ε, for $\varepsilon = 0.001$ or 0.01. Delta and gamma are estimated as numerical derivatives using the option values obtained from the new trees in Eq. (20). "Tree extension" involves building the valuation tree starting one time step before time 0, so that it produces option values at time 0 for asset prices X_0, $X_0 + h$, and $X_0 - h$, as shown in Fig. 7. These are then used in Eq. (20) to calculate numerical derivatives.

$$\Delta = \frac{\partial C}{\partial S} = \frac{\partial C}{\partial \ln(S)}\frac{1}{S} \approx \frac{C(X_0+\varepsilon) - C(X_0-\varepsilon)}{2\varepsilon}\frac{1}{S},$$

$$\Gamma = \frac{\partial \Delta}{\partial S} = \frac{\partial^2 C}{\partial S^2} = \frac{\partial}{\partial S}\left(\frac{\partial C}{\partial \ln(S)}\frac{1}{S}\right) = \left(\frac{\partial^2 C}{\partial (\ln(S))^2} - \frac{\partial C}{\partial \ln(S)}\right)\frac{1}{S^2}$$

$$\approx \left(\frac{C(X_0+\varepsilon) + C(X_0-\varepsilon) - 2C(X_0)}{\varepsilon^2} - \frac{C(X_0+\varepsilon) - C(X_0-\varepsilon)}{2\varepsilon}\right)\frac{1}{S^2}. \quad (20)$$

The models are based on an initial stock price of 40, a riskless interest rate of 5.0% (4.88% continuously compounded), and no dividends. The option test set includes European puts with all 27 combinations of: strike prices of 35, 40, and 45; maturities of 1, 4, and 7 months; and volatilities of 0.20, 0.30, and 0.40. Root mean squared errors relative to the exact Black–Scholes values and computation times on a Dell Pentium Pro 200 MHz computer are displayed for the same test set of 27 European put options as in Table 1. The execution time is the average from 10 identical runs for each set of parameters. N is the number of time steps to maturity.

Model	Approximation RMSE			Execution time (s)
	Price	Delta	Gamma	
$N = 25$				
Trinomial perturbation, $\varepsilon = 0.001$	0.012025	0.034958	0.499567	0.0200
Trinomial perturbation, $\varepsilon = 0.01$	0.012025	0.020866	0.101378	0.0200
Trinomial tree extension	0.012025	0.003337	0.000428	0.0100
$N = 100$				
Trinomial perturbation, $\varepsilon = 0.001$	0.002770	0.015642	0.242340	0.2810
Trinomial perturbation, $\varepsilon = 0.01$	0.002770	0.006151	0.044298	0.2800
Trinomial tree extension	0.002770	0.000846	0.000144	0.0920
$N = 250$				
Trinomial perturbation, $\varepsilon = 0.001$	0.001360	0.009286	0.242539	1.5820
Trinomial perturbation, $\varepsilon = 0.01$	0.001360	0.001689	0.026351	1.5830
Trinomial tree extension	0.001360	0.000346	0.000061	0.5298
$N = 1000$				
Trinomial perturbation, $\varepsilon = 0.001$	0.000244	0.004631	0.120938	24.6160
Trinomial perturbation, $\varepsilon = 0.01$	0.000244	0.000656	0.004602	24.6650
Trinomial tree extension	0.000244	0.000079	0.000015	8.1918

formula, as well as execution time on a Pentium Pro 200 MHz personal computer (averaged over 10 runs for each set of parameters). Results are shown for 25, 100, 250 and 1000 step trees.

Since the procedure for calculating Greek letter exposures does not change the original valuation tree, the RMSE for the option price is the same for all variants. The first two lines for each value of N use perturbation to compute numerical derivatives. Building two additional complete trees should take at least three times as long as simply pricing the option. We also see an interesting effect with regard to the size of the perturbation. Using a very small value, $\varepsilon = 0.001$, produces much less accurate deltas and, especially, gammas than using a larger value, $\varepsilon = 0.01$. This is because dividing by a very small ε in taking the numerical derivatives magnifies the effect of the nonlinearity error. Thus, it can be counterproductive to use very small perturbations in estimating the Greek letter risk exposures. The third line gives results for the method of extending the tree backwards in time. This is considerably more efficient in terms of computation time than the perturbation approaches, and it also gives much more accurate results, especially for gamma.

Unfortunately, the problem of nonlinearity error still remains. This can partly be mitigated by the use of an AMM model to add a section of fine mesh around the strike price at expiration, as demonstrated in Section 3. But there is still a difficulty at time 0, because to compute delta and gamma one must either solve three complete lattices with a small ε perturbation or else extend the tree and compute them from nonmarginal asset price changes.

5.3. *Building an AMM model to compute delta and gamma*

This section shows how an AMM model with a region of fine mesh around the initial asset price allows numerical derivatives to be computed using as small an ε perturbation as we like with only a minor increase in the total number of node calculations. Fig. 8 shows a lattice set up much like the one in Fig. 7, with perturbation trees that overlap the original one and add new nodes only at the highest and lowest prices in each period. The difference here is that these new sections of lattice begin at time 0 at asset prices of $X_0 + h/2$ and $X_0 - h/2$, that is, at deviations from the original price X_0 that are half as large as in the extended tree of Fig. 7.

To do this, we introduce quadrinomial branching for these two nodes. In the original lattice, starting from X_0 there are paths going up to $X_0 + h$, down to $X_0 - h$, and in between to remain at X_0 in the next period. The extended tree would add a time 0 node at the price $X_0 + h$, from which branches would lead to $X_0 + 2h$, $X_0 + h$, and X_0. The quadrinomial branching for the new node we are placing at $X_0 + h/2$ allows for price changes to any of the four second period nodes that could be reached from either X_0 or $X_0 + h$. The modified tree cuts the perturbation used in the delta and gamma calculation in half, but requires

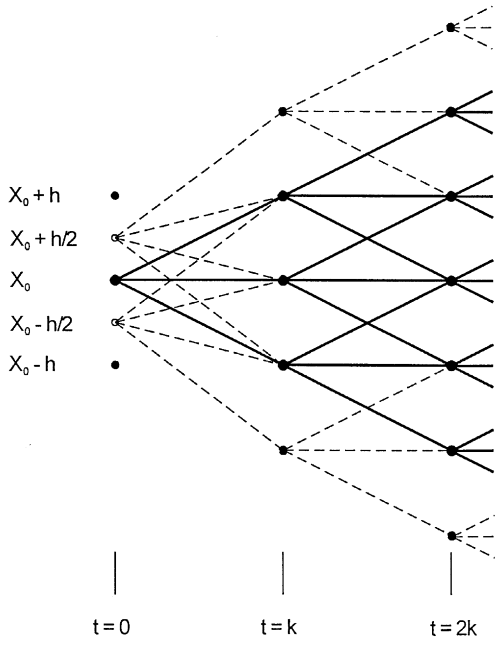

Fig. 8. AMM level 1 for computing delta and gamma. The trinomial tree is extended with a region of fine mesh around the initial asset price. The dotted lines indicate the new sections of lattice to be added only at the highest and lowest prices in each time step. The new sections of lattice begin at time 0 at asset prices $X_0 + h/2$ and $X_0 - h/2$, that is, at deviations that are half as large as those in the remainder of the tree. Four branches attach each of these new $t = 0$ nodes to nodes in the coarse lattice with price steps of h.

exactly the same number of new nodes as the extended tree of Fig. 7 because it connects to the original lattice in exactly the same way at the end of the first time step. We use quadrinomial branching to attach the new nodes to the extended tree because trinomial branching from a node like $X_0 + h/2$, which is not at a price for which there are nodes in the coarse tree, can produce negative probabilities.

Quadrinomial branching introduces a fourth probability to be determined. Consider the new node at $X_0 + h/2$. The four probabilities attached to the four possible next period nodes are p_{uu}, p_u, p_d, and p_{dd}, corresponding to price changes of $+\frac{3}{2}h$, $+h/2$, $-h/2$, and $-\frac{3}{2}h$, respectively. Eq. (5) specifies four conditions for a trinomial system to satisfy, which fixes the three probabilities and the relationship between the price and time steps h and k. Here, the price and time steps in the new fine mesh are already determined by their values in the coarse mesh, so adding a fourth branch does not gain a degree of freedom.

The four probabilities must obey the following conditions: the expected return over the next time step is the riskless rate, the second moment is consistent with

the volatility σ of the underlying asset, the skewness is zero like that of the normal distribution, and the probabilities sum to 1.0. With $\alpha = r - q - \sigma^2/2$ and q denoting the dividend yield,

$$\begin{aligned}
&\tfrac{3}{2}hp_{uu} + \tfrac{1}{2}hp_u - \tfrac{1}{2}hp_d - \tfrac{3}{2}hp_{dd} = 0,\\
&(\tfrac{3}{2}h)^2p_{uu} + (\tfrac{1}{2}h)^2p_u + (\tfrac{1}{2}h)^2p_d + (\tfrac{3}{2}h)^2p_{dd} = \sigma^2k,\\
&(\tfrac{3}{2}h)^3p_{uu} + (\tfrac{1}{2}h)^3p_u - (\tfrac{1}{2}h)^3p_d - (\tfrac{3}{2}h)^3p_{dd} = 0,\\
&p_{uu} + p_u + p_d + p_{dd} = 1. \qquad (21)
\end{aligned}$$

Solving these four equations in four unknowns, making use of the fact that $h^2 = 3\sigma^2k$, and simplifying yields

$$\begin{aligned}
&p_{uu} = p_{dd} = 1/48,\\
&p_u = p_d = 23/48. \qquad (22)
\end{aligned}$$

5.4. *Adding finer AMM levels to the lattice*

Like the other AMM lattice structures, this model is isomorphic, so finer layers of mesh can be added using the same procedure at each level. Fig. 9 illustrates how the second level AMM is constructed. The first level of the AMM, shown in Fig. 8, begins with the extended-tree lattice, using a price step of h and a time step of k, and adds time 0 nodes one half of a price step above and below X_0. The time 0 nodes are connected to the coarse lattice at time 1, using trinomial branching from X_0, since it is a node in the coarse lattice, and quadrinomial branching from the two new nodes that fall in between two coarse nodes. The final tree has three nodes at time 0, whose prices differ by the amount $h/2$. From these nodes there are branches connecting to the five time 1 nodes in a coarse tree with price step h.

To go to the next level of AMM, we must add a new section of lattice with a price step of $h/2$ and a time step of $k/4$ and connect it to five nodes in the first level AMM lattice. The initial asset price X_0 is the center node at $t = 0$. From it, three branches lead to the nodes one (fine mesh) time step later, with prices X_0, $X_0 + h/2$, and $X_0 - h/2$. As in the level 1 AMM, we now place time 0 nodes one half of a (fine mesh) price step above and below X_0, at $X_0 + h/4$ and $X_0 - h/4$. From these prices, a middle "no change" branch would not connect to the next higher level lattice, so quadrinomial branching is required. As before, branches from the three initial price nodes lead to five nodes of the next higher level mesh in the next (fine) time step. (This means that the coarse tree nodes at $X_0 + h$ and $X_0 - h$, which did not have to be calculated for the first level AMM, will be needed now.)

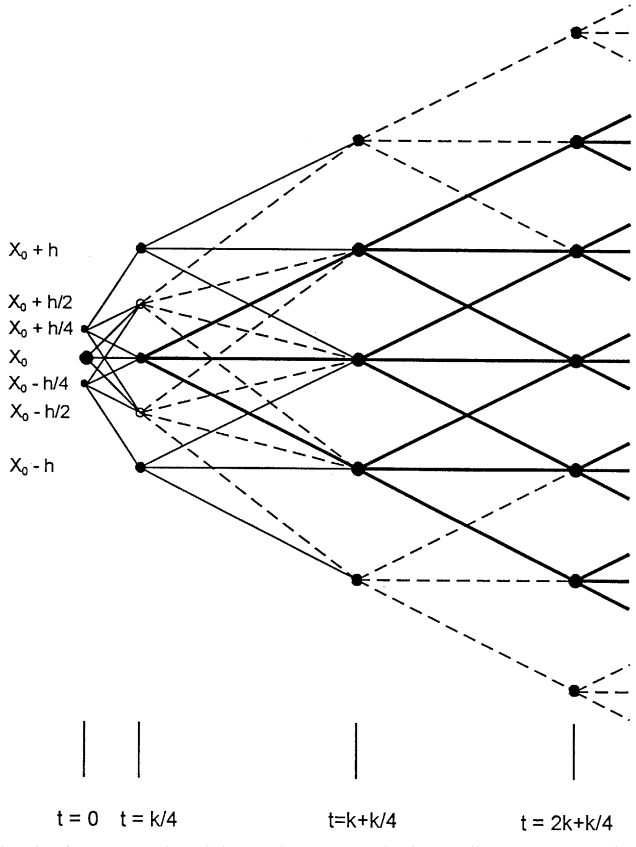

Fig. 9. AMM level 2 for computing delta and gamma. The heavy lines represent the coarse lattice with step sizes h and k. The dashed lines indicate the section of lattice that is added in the first level AMM. The fine solid lines show the level 2 AMM section, with a price step of $h/2$ and a time step of $k/4$. New time 0 nodes are at $X_0 + h/4$ and $X_0 - h/4$, with four branches extending from each. As before, three branches extend from X_0.

In Fig. 9, the heavy lines represent the coarse lattice with step sizes h and k; the dashed lines indicate the section of lattice that is added in the first level AMM, and the fine solid lines show the level 2 AMM section. Adding a third level AMM would follow the same process, with the starting node one finest time step ($k/16$) earlier and three $t = 0$ nodes at X_0, $X_0 + h/8$, and $X_0 - h/8$, that branch to five nodes in the next step. In all cases, if there is a node with the same price in the next higher AMM level, trinomial branching is used, but for nodes falling in between prices in the next level, branching is quadrinomial.

5.5. The general procedure for constructing an AMM lattice for delta and gamma

The foregoing description started with the coarsest mesh lattice and considered adding progressively finer sections, but in constructing this kind of AMM from scratch, it is actually more intuitive to think about it starting from time 0. Suppose we decide to build a tree with N coarse steps and M AMM levels. Note that the first level does not require adding any mesh with a shorter time step, as is clear from Fig. 8. For each AMM level $m > 1$, there will be one time step of length $k/4^{m-1}$. Thus the length of the entire tree will be $(N + 1/4 + \cdots + 1/4^{M-1})$ coarse time steps. Given option maturity T and volatility σ, the values for h and k are given by

$$k = \frac{T}{N - 1 + \sum_{m=1}^{M} 1/4^{m-1}},$$

$$h = \sigma\sqrt{3k}. \tag{23}$$

Denote the price and time steps for the mth level of the AMM as h_m and k_m ($h_m = h/2^{m-1}$, $k_m = k/4^{m-1}$). The first node is placed at X_0 at time 0 and the nodes to be used in the calculation of delta and gamma are placed at $X_0 + h_m/2$ and $X_0 - h_m/2$. These three nodes must branch one k_m time step later to five nodes in the next higher level AMM lattice, located at $X_0 + 2h_m$, $X_0 + h_m$, X_0, $X_0 - h_m$, and $X_0 - 2h_m$. The branching from X_0 is trinomial, because there is a node in the higher level lattice at X_0. Branching from a node lying between two higher level lattice prices is quadrinomial. Once the nodes and branching for the first k_m time step are completed, the price step is doubled and the time step multiplied by four, and the process of constructing the nodes and branching to the next higher level is repeated until the coarsest tree with price and time steps h and k is reached. This coarse lattice is then extended out to option maturity.

5.6. Performance of the AMM in estimating delta and gamma

Table 5 reports performance statistics for the standard trinomial and for several versions of AMM models. We examined adding fine mesh at time 0 to improve the estimates of delta and gamma. However, these numerical derivatives are affected by the nonlinearity error in the option prices, so that accuracy can be enhanced considerably by using an AMM at expiration ($t = T$) as well. For each N, the first line of results is for the standard trinomial, with the Greek letters computed by extending the tree. The next two lines incorporate one and two levels of AMM at time 0, as in Figs. 8 and 9. The last three lines add AMM levels both at the beginning and at expiration.

In principle, adding finer lattice only at the beginning should not affect the calculation of the option value, but there is a small effect going from AMM 1 to AMM 2. This is due to the slight change in the time step, e.g., changing from 25

Table 5
Performance of trinomial AMM models in estimating delta and gamma for European puts

The table compares the performance of trinomial AMM models with different levels of fine mesh added at $t = 0$ around the initial asset price, and at $t = T$ around the strike price. The first line in each panel gives the results for the standard Trinomial model. Delta and gamma are estimated as numerical derivatives using the option values obtained from the finest level of mesh at $t = 0$. Root mean squared errors relative to the exact Black–Scholes values and computation times on a Dell Pentium Pro 200 MHz computer are displayed for a test set of 27 European put options. The models are based on an initial stock price of 40, a riskless interest rate of 5.0% (4.88% continuously compounded), and no dividends. The option test set includes European puts with all 27 combinations of: strike prices of 35, 40, and 45; maturities of 1, 4, and 7 months; and volatilities of 0.20, 0.30, and 0.40. The execution time is the average from 10 identical runs for each set of parameters.

	AMM level		Approximation RMSE			Execution time (s)
	$t = 0$	$t = T$	Price	Delta	Gamma	
$N = 25$						
Trinomial	0	0	0.012025	0.003337	0.000428	0.0090
AMM	1	0	0.012025	0.001087	0.000333	0.0090
	2	0	0.011909	0.000810	0.000400	0.0090
	1	1	0.002812	0.000845	0.000080	0.0120
	2	2	0.000596	0.000205	0.000113	0.0131
	3	3	0.000193	0.000053	0.000120	0.0140
$N = 100$						
Trinomial	0	0	0.002770	0.000846	0.000144	0.0931
AMM	1	0	0.002770	0.000265	0.000068	0.0922
	2	0	0.002764	0.000188	0.000077	0.0931
	1	1	0.000600	0.000210	0.000020	0.0942
	2	2	0.000153	0.000056	0.000028	0.0971
	3	3	0.000043	0.000014	0.000027	0.0991
$N = 250$						
Trinomial	0	0	0.001360	0.000346	0.000061	0.5358
AMM	1	0	0.001360	0.000115	0.000029	0.5288
	2	0	0.001350	0.000085	0.000031	0.5287
	1	1	0.000245	0.000079	0.000006	0.5308
	2	2	0.000057	0.000023	0.000010	0.5878
	3	3	0.000019	0.000005	0.000011	0.5738
$N = 1000$						
Trinomial	0	0	0.000244	0.000079	0.000015	8.5072
AMM	1	0	0.000244	0.000021	0.000006	8.3711
	2	0	0.000243	0.000016	0.000007	8.3640
	1	1	0.000056	0.000023	0.000002	8.3170
	2	2	0.000016	0.000005	0.000003	8.3130
	3	3	0.000006	0.000001	0.000003	8.3751

to $25\frac{1}{4}$ coarse steps in the tree. The big difference is in the accuracy of delta and gamma, for which RMSE is cut by about 2/3 for delta and 1/2 for gamma in the AMM 1 model, with no increase in execution time. Going from AMM 1 to AMM 2 improves the delta calculation a little, but not gamma.

When we add finer mesh both at the beginning and the end, there is a sharp improvement in both the option value and the delta. Gamma also becomes much more accurate when the first level of AMM mesh is added at expiration, but further refinement does not seem to help. If anything, using a single level of fine mesh is better than using more.[14] It is important to notice that while adding finer mesh leads to a large improvement in accuracy, execution time is virtually the same for the most refined AMM model as for the basic Trinomial. Finally, in comparing the AMM approach to the other methods displayed in Table 5, it is apparent that huge gains in performance are possible. For example, a 25 time step AMM model with three levels of fine mesh at the beginning and end achieves substantially greater accuracy in pricing the options and computing their deltas than does a 1000 time step standard Trinomial that takes more than 500 times longer to execute. Only in estimating gamma does the AMM seem to have any difficulty: it improves accuracy by *only* about a factor of 5 in the same amount of time as a standard model.

6. Conclusion

The significant advance in the technology of option valuation achieved by the Black–Scholes pricing model quickly reached an impasse with American options and other derivative instruments with more varied contingencies. Although the principles of valuation based on the no-arbitrage condition continue to hold, closed-form equations can not be derived. Lattice-based models, starting with the Binomial and followed by the Trinomial, offer an intuitive framework for obtaining approximate solutions.

Numerous enhancements to tree-based models have led to some improvements in performance over the years, but whole classes of problems, including many that are important for pricing common real-world derivative instruments, remain theoretically soluble but practically infeasible with the standard methods because they require too many calculations to achieve an acceptable degree of accuracy.

[14] This is a result that we have so far been unable to explain satisfactorily. Holding the number of AMM levels at expiration constant, we find that the first layer of adaptive mesh improves the accuracy of the gamma calculation much more than expected (e.g., in a 100 step tree with 10 AMM layers at expiration, adding one AMM layer at the beginning cuts RMSE for gamma from 0.000143 to 0.000005). With the second layer, performance degrades, but then it improves gradually as further layers are added.

The adaptive mesh approach described in this paper offers a way to increase accuracy and reduce computation time enormously for many sorts of valuation problems. We have described three types of AMM, one that involves constructing a small section of high resolution mesh around the strike price at expiration, a second that creates one or more layers of fine mesh near the barrier of a barrier option, and finally one that adds fine mesh at time 0 to enhance estimation of delta and gamma. All three AMM models produce major improvements in accuracy and computation speed.

The AMM approach can also be adapted relatively easily to higher dimensions, with even greater performance increases relative to constant mesh size lattice methods. For example, we obtain some results using an AMM procedure to value an "outside barrier option". This is a barrier option for which the payoff is determined by one asset price while the barrier depends on a different one. An option on the British FT-SE stock index that is knocked out if the exchange rate on the pound falls below a specified level is one example. With American exercise and the starting asset price near the knock-out barrier, these valuation problems are practically insoluble with ordinary lattice methods (or any other technique currently in use). The AMM technology therefore greatly extends the range of derivative valuation problems that may be addressed.[15]

Along with developing AMM models for use with derivatives that are contingent upon multiple stochastic factors, we are also exploring other "variations on the theme". This includes developing models to deal with curved or discontinuous barriers and exploring general criteria for determining in which regions of an option's state space an AMM approach will be most valuable.

Appendix A

This Appendix proves that as the sizes of the time and price steps go to zero, the option value obtained from the AMM non-standard branching setup for barrier options, as described in Section 4, converges to the true value. True value is defined as the value of a continuous-time continuous-state barrier option that would be obtained by solving the fundamental partial differential equation of contingent claims, with the appropriate boundary conditions. The same kind of proof can easily be adapted to prove convergence for other AMM structures.

The proof is in the same spirit as a consistency proof for the explicit finite difference method of solving partial differential equations (see Fletcher, 1991, p. 77). The true solution for the option value satisfies the PDE and boundary conditions exactly, while the approximation satisfies them only up to an error.

[15] Gao (1996) discusses an outside barrier option example in which the RTM approach would require over 100 days of CPU time (it is estimated), while an AMM model obtains the correct price to 3 decimal places in under 1 s.

The proof consists of showing that the error term goes to zero as the time step goes to zero.

The barrier option price must satisfy the following PDE and boundary conditions:

$$\begin{aligned} &C_t + \tfrac{1}{2}\sigma^2 C_{xx} + \alpha C_x - rC = 0, \\ &C(x, T) = (\mathrm{e}^x - K)^+ \ \ \forall x, \\ &C(\ln H, t) = 0 \ \ \forall 0 < t \leqslant T, \end{aligned} \tag{A.1}$$

where the variables have been defined previously, subscripts denote partial derivatives, and $(.)^+$ denotes the value of the expression in parentheses when it is positive and 0 otherwise.

In a trinomial model for a barrier option, placing a layer of nodes on the barrier causes the approximation to satisfy the above boundary conditions exactly. A trinomial valuation equation $C^{\mathrm{A}}(x, t)$ can be written as follows:

$$\begin{aligned} C^{\mathrm{A}}(x, t) &= \mathrm{e}^{-rk}[p_{\mathrm{u}}(h, k)C^{\mathrm{A}}(x + h, t + k) + p_{\mathrm{m}}(h, k)C^{\mathrm{A}}(x, t + k) \\ &\quad + p_{\mathrm{d}}(h, k)C^{\mathrm{A}}(x - h, t + k)], \\ C^{\mathrm{A}}(x, T) &= (\mathrm{e}^x - K)^+, \ \ \forall x, \\ C^{\mathrm{A}}(\ln H, t) &= 0, \ \ \forall 0 < t < \ \leqslant T. \end{aligned} \tag{A.2}$$

The exact solution to the barrier option problem satisfies the PDE (A.1). We rewrite the trinomial solution (A.2) in the form of (A.1) and evaluate the discrepancy. First, we expand the right-hand side of (A.2) in Taylor series around the value $C^{\mathrm{A}}(x, t)$. We show all terms in the expansion out to order k^2, keeping in mind that the price step h is proportional to the square root of the time step, $h \propto k^{1/2}$:

$$\begin{aligned} C^{A}(x, t) &= \mathrm{e}^{-rk}[p_{\mathrm{u}}(h, k)C^{\mathrm{A}}(x + h, t + k) + p_{\mathrm{m}}(h, k)C^{\mathrm{A}}(x, t + k) \\ &\quad + p_{\mathrm{d}}(h, k)C^{\mathrm{A}}(x - h, t + k)] \\ &= \mathrm{e}^{-rk}\Bigg\{\tfrac{1}{2}\left(\frac{\sigma^2 k}{h^2} + \frac{\alpha^2 k^2}{h^2} + \frac{\alpha k}{h}\right)[C + C_x h + C_t k \\ &\quad + \tfrac{1}{2}C_{xx}h^2 + \tfrac{1}{2}C_{tt}k^2 + C_{xt}hk \\ &\quad + \tfrac{1}{6}C_{xxx}h^3 + \tfrac{1}{2}C_{xxt}h^2 k + \tfrac{1}{24}C_{xxxx}h^4 + \mathrm{o}(k^2)] \\ &\quad + \tfrac{1}{2}\left(\frac{\sigma^2 k}{h^2} + \frac{\alpha^2 k^2}{h^2} - \frac{\alpha k}{h}\right)[C - C_x h + C_t k + \tfrac{1}{2}C_{xx}h^2 + \tfrac{1}{2}C_{tt}k^2 \\ &\quad - C_{xt}hk - \tfrac{1}{6}C_{xxx}h^3 + \tfrac{1}{2}C_{xxt}h^2 k + \tfrac{1}{24}C_{xxxx}h^4 + \mathrm{o}(k^2)] \\ &\quad + \left(1 - \left(\frac{\sigma^2 k}{h^2} + \frac{\alpha^2 k^2}{h^2}\right)\right)[C + C_t k + \tfrac{1}{2}C_{tt}k^2 + \mathrm{o}(k^2)]\Bigg\}. \end{aligned} \tag{A.3}$$

The term e^{-rk} can also be expanded as an infinite series as:

$$e^{-rk} = 1 - rk + \tfrac{1}{2}r^2k^2 + o(k^2).$$

Multiplying through in Eq. (A.3) and gathering terms of $o(k^2)$ into the remainder, gives

$$\begin{aligned} C^A = C^A &+ (C_t^A + \tfrac{1}{2}\sigma^2 C_{xx}^A + \alpha C_x^A - rC^A)k \\ &- (C_t^A + \tfrac{1}{2}\sigma^2 C_{xx}^A + \alpha C_x^A - rC^A)rk^2 \\ &+ (\tfrac{1}{2}r^2 C^A k^2 + \tfrac{1}{2}\alpha^2 C_{xx}^A k^2 + \tfrac{1}{2}\sigma^2 C_{xxt}^A k^2 \\ &+ \tfrac{1}{24}\sigma^2 C_{xxxx}^A h^2 k + \tfrac{1}{2}C_{tt}^A k^2 + \alpha C_{xt}^A k^2 + \tfrac{1}{6}\alpha C_{xxx}^A h^2 k) + o(k^2). \end{aligned} \tag{A.4}$$

Subtracting C^A from both sides and dividing through by k reduces the trinomial pricing equation to the following PDE with an error term:

$$C_t^A + \tfrac{1}{2}\sigma^2 C_{xx}^A + \alpha C_x^A - rC^A + \varepsilon^A = 0, \tag{A.5}$$

where the error term is

$$\begin{aligned} \varepsilon^A = &- (C_t^A + \tfrac{1}{2}\sigma^2 C_{xx}^A + \alpha C_x^A - rC^A)rk + (\tfrac{1}{2}r^2 C^A k + \tfrac{1}{2}\alpha^2 C_{xx}^A k + \tfrac{1}{2}\sigma^2 C_{xxt}^A k \\ &+ \tfrac{1}{24}\sigma^2 C_{xxxx}^A h^2 + \tfrac{1}{2}C_{tt}^A k + \alpha C_{xt}^A k + \tfrac{1}{6}\alpha C_{xxx}^A h^2) + o(k) \propto O(k). \end{aligned} \tag{A.6}$$

It is clear in Eq. (A.6) that ε^A goes to zero as k goes to zero. Therefore, in the limit the trinomial approximation obeys the PDE with boundary conditions (A.1). This proves that the basic trinomial pricing equation for a barrier option is consistent.

We now apply the same approach to our AMM barrier option model. Since the A level lattice (see Fig. 6) is just a standard trinomial, by the above analysis it is consistent at all points. Now consider the points in the B level mesh. Although there are several sets of branch probabilities for the different types of B-level nodes, the branching from any given node is always in the standard trinomial form. The time step is proportional to k^A (from the A level mesh) and the price step is proportional to the square root of the time step. Thus Eqs. (A.3)–(A.6) are equally valid for a B level node. We have

$$\begin{aligned} &\varepsilon^A \propto k^A, \\ &\varepsilon^B \propto k^B \propto k^A. \end{aligned} \tag{A.7}$$

So as k^A goes to zero, the error terms also go to zero at all A and B level nodes.

The nonstandard AMM branching is therefore consistent. A similar argument can easily show that the AMM pricing equation for a finer mesh size is similar to that of the level B mesh and is also consistent.

References

Black, F., Scholes, M., 1973. The pricing of options and corporate liabilities. Journal of Political Economy 81, 637–654.

Boyle, P., Lau, S., 1994. Bumping up against the barrier with the binomial method. The Journal of Derivatives 1, 6–14.

Brennan, M., Schwartz, E., 1977. The valuation of American put options. Journal of Finance 32, 449–462.

Broadie, M., Detemple, J., 1996. American option valuation: new bounds, approximations, and a comparison of existing methods. Review of Financial Studies 9, 1211–1250.

Canina, L., Figlewski, S., 1993. The informational content of implied volatility. Review of Financial Studies 6, 659–681.

Cho, H., Lee, K., 1995. An extension of the three-jump process model for contingent claim valuation. The Journal of Derivatives 3, 102–108.

Cox, J., Ross, S., Rubinstein, M., 1979. Option pricing: A simplified approach. Journal of Financial Economics 7, 229–264.

Fletcher, C., 1991. Computational Techniques for Fluid Dynamics: Fundamental and General Techniques. Vol. 1. Springer, Berlin.

Gao, B., 1996. Essays in efficient option pricing. Unpublished Ph.D. Dissertation. New York University Stern School of Business, New York.

Gastineau, G., Kritzman, M., 1996. Dictionary of Financial Risk Management. Frank J. Fabozzi Associates, New Hope, Pennsylvania.

Geske, R., Johnson, H., 1984. The American put option valued analytically. Journal of Finance 39, 1511–1524.

Hull, J., White, A., 1990. Valuing derivative securities using the explicit finite difference method. Journal of Financial and Quantitative Analysis 25, 87–100.

Merton, R., 1973. Theory of rational option pricing. Bell Journal of Economics and Management Science 4, 141–183.

Pelsser, A., Vorst, T., 1994. The binomial model and the Greeks. The Journal of Derivatives 1, 45–49.

Ritchken, P., 1995. On pricing barrier options. The Journal of Derivatives 3, 19–28.

Rubinstein, M., 1991. Exotic options. Berkeley Research Program in Finance Working Paper No. 220, University of California at Berkeley.