The Journal of Financial Research • Vol. XXXI, No. 4 • Pages 381–408 • Winter 2008

ADAPTIVE MESH MODELING AND BARRIER OPTION PRICING UNDER A JUMP-DIFFUSION PROCESS

Michael Albert
Duke University

Jason Fink and Kristin E. Fink
James Madison University

Abstract

The computational burden of numerical barrier option pricing is significant, even prohibitive, for some parameterizations—especially for more realistic models of underlying asset behavior, such as jump diffusions. We extend a binomial jump diffusion pricing algorithm into a trinomial setting and demonstrate how an adaptive mesh may fit into the model. Our result is a barrier option pricing method that employs fewer computational resources, reducing run times substantially. We demonstrate that this extension allows the pricing of options that were previously computationally infeasible and examine the parameterizations in which use of the adaptive mesh is most beneficial.

JEL Classification: G13, G19

I. Introduction

Substantial progress has been made in developing empirically valid derivatives pricing models, primarily by allowing for more realistic behavior of the underlying asset, such as the Press (1967) jump-diffusion model. Although Merton (1976) first priced European options where the underlying asset followed this process, lattice structures allowing convergent approximations to the values of American options have been only recently presented. Though appropriately specified binomial trees and similar models do converge to the correct values for barrier options when the underlying follows a jump process, accurate convergence may take a prohibitively long time. We extend the adaptive mesh model of Figlewski and Gao (1999a) to the jump-diffusion binomial framework of Hilliard and Schwartz (2005), thereby improving the computational efficiency of the solution method for the pricing

The authors would like to thank Gerald Gay (the editor), Ajay Subramanian (the referee), Joseph Albert, Mark Bertus, Harry Reif, and Young Choi, as well as participants in the James Madison University finance workshop and the 2006 Southern Finance Association meetings.

of barrier options. We achieve a 60-fold decrease in the run time of the model for close-to-the-barrier knockout options and price some options for which lattice solutions were previously unattainable. This study contributes to the literature both by demonstrating this extension and by revealing a relation that describes under which circumstances the adaptive mesh procedure is most valuable. In particular, the adaptive mesh implementation is most beneficial when the stock price is close to the barrier, when the volatility of the "smooth" component of the jump process is large, and when there is a long time until the expiration of the option.

The jump-diffusion model allows for a density of log asset returns that better resembles the skewness and kurtosis observed in some markets. Amin (1993) and Hilliard and Schwartz (2005) present lattice models capable of pricing a wide variety of options on this process. The model in Hilliard and Schwartz can effectively accommodate a wide variety of parameter specifications, notably providing greater accuracy than previously possible when the volatility of the jump is high. The authors demonstrate the effectiveness of their bivariate binomial technique for approximating the price of European and American options, as well as the associated hedging parameters. They leave inquiry into exotic options, such as barrier options, for future research.

Continuously monitored barrier options, however, have proven difficult to price accurately via a binomial method, even in a simpler geometric Brownian motion (GBM) setting (Boyle and Lau 1994).[1] The pricing of barrier options is an important issue, however. There is a large market for barrier options, notably in currencies. Furthermore, Brockman and Turtle (2003) find evidence that down-and-out calls may be the appropriate way to value corporate equity, and Broadie and Kaya (2007) present a binomial model where the pricing of debt has barrier option characteristics.

Even with simple characterizations of the underlying asset, the run time necessary for lattice models to accurately converge to barrier option values may render the models not useful, or even impossible, to implement. For example, Figlewski and Gao (1999a) demonstrate that the implementation of one efficiency-enhancing modification (provided by Ritchken 1995) with a reasonable parameterization (in a world where the stock follows GBM) requires a trinomial tree of almost 100,000 time steps to price a down-and-out call option. Figlewski and Gao introduce adaptive mesh modeling to address this difficulty. They demonstrate a dramatic reduction in computational time using this technique.

We demonstrate that the computational cost of similar problems are even greater in a setting such as that of Hilliard and Schwartz (2005). However, the

[1]When the underlying asset follows GBM, Andricopoulos et al. (2003, 2007) provide a technique (referred to as QUAD) to value discretely monitored barrier options based on quadrature methods. The effectiveness of the procedure when the barrier is continuously monitored has not been studied and is a situation in which the QUAD procedure is likely to exhibit significantly reduced efficiency, and neither has that method been explored in the context of jump processes.

extension of the adaptive mesh into a jump-diffusion setting is not immediately apparent. We present an extension of the adaptive mesh technology for the pricing of continuously monitored barrier options to the jump-diffusion pricing model of Hilliard and Schwartz. Finally, we analyze the circumstances in which this extension is most useful to the practitioner.

II. The State of Barrier Option Pricing

The difficulty of pricing barrier options with standard lattice techniques is well established in the literature. Consider the standard assumption of GBM where the underlying asset follows the process

$$\frac{dS}{S} = (r - d)\,dt + \sigma\,dz, \tag{1}$$

where S is the price of the underlying asset, dS is changes in this asset price, r is the continuously compounded rate of interest, d is a proportional dividend yield continuously paid by the underlying asset, σ is a volatility parameter, dt is the change in time, and dz is standard Brownian motion. When this process determines fluctuations in the asset price, the price of many exotic options have quasi-closed-form solutions. Because of this, the GBM assumption provides a convenient laboratory to examine lattice methods.

The crux of the pricing difficulty concerning barrier options is the severe nonlinearity between the value of the option on the barrier and its value when the underlying asset approaches but does not cross the barrier. As Boyle and Lau (1994) demonstrate, this feature creates a barrier option price convergence pattern that exhibits wide swings from the true price as time steps are increased on a binomial tree. In the context of trinomial trees, Ritchken (1995) demonstrates this same phenomenon and provides a method to help mitigate the problem.

Beginning with an initial log asset price in a standard trinomial tree, upticks and downticks are taken to be symmetric. Therefore at time Δt an uptick yields an asset price of $\ln(S) + h$, and a downtick yields a value of $\ln(S) - h$. The middle tick leaves the asset price unchanged. Here, we follow the method of Ritchken (1995) and set the magnitude of h outside of the model. The tick probabilities are determined by solving a set of moment-matching equations that restrict the summation of the probabilities to equal 1. We therefore have

$$\begin{aligned}
&E(\ln(S(t + \Delta t) - \ln(S(t)) = \left(r - d - \frac{\sigma^2}{2}\right)\Delta t = p_u h + p_m 0 + p_d(-h) \\
&E([\ln(S(t + \Delta t) - \ln(S(t)]^2) = \sigma^2 \Delta t = p_u h^2 + p_m 0 + p_d(-h)^2 \\
&1 = p_u + p_m + p_d,
\end{aligned} \tag{2}$$

and h given. The solution to these equations yields:

$$p_u = \frac{\left(r - d - \frac{\sigma^2}{2}\right)\Delta t}{2h} + \frac{\sigma^2 \Delta t}{2h^2}$$
$$p_m = 1 - p_u - p_d \tag{3}$$
$$p_d = \frac{\sigma^2 \Delta t}{2h^2} - \frac{\left(r - d - \frac{\sigma^2}{2}\right)\Delta t}{2h}$$

Substitution for p_m in equation (3) gives,

$$p_m = 1 - \frac{\sigma^2(\Delta t)}{h^2}. \tag{4}$$

We may eliminate the possibility of negative probabilities by defining a free parameter $\Lambda > 1$ such that

$$\Lambda = \frac{h^2}{\sigma^2(\Delta t)}, \tag{5}$$

and allow this equation to implicitly set the relationship between h and Δt. Evidence indicates that $\Lambda = 3$ provides nice convergence properties for trinomial trees, and we follow Figlewski and Gao (1999a) in assuming this value for subsequent examples.

The Ritchken (1995) contribution is to demonstrate that choosing h to produce an integer number of time steps between the log asset price and barrier price can improve the trinomial algorithm in pricing barrier options (this selection requires small modification in Λ for Δt to be such that time may be discretized into an integer number of time steps). As Figlewski and Gao (1999a) demonstrate, the effectiveness of the Ritchken technique reaches a computational boundary as the asset price approaches the barrier. Consider the pricing of a down-and-out call with a knock-out barrier of 50 with the following parameters: $S = 50.15$, $r = .05$, $d = 0$, and $\sigma = .25$. Following the logic of Figlewski and Gao, the largest h that permits a single downtick before hitting the barrier is $h = \ln(50.15) - \ln(50) = .002996$. For $\Lambda = 3$, this implies a Δt of .000048, or 20,889 time steps in the tree—possibly a prohibitive computation in the time frame of a financial decision (see Figlewski and Gao 1999b for further discussion).

This computational problem is rectified by Figlewski and Gao's (1999a) adaptive mesh model. Their insight is that a reduction in computational time is attained by pricing the option on a relatively coarse lattice and then grafting a smaller, finer tree onto this coarser lattice.

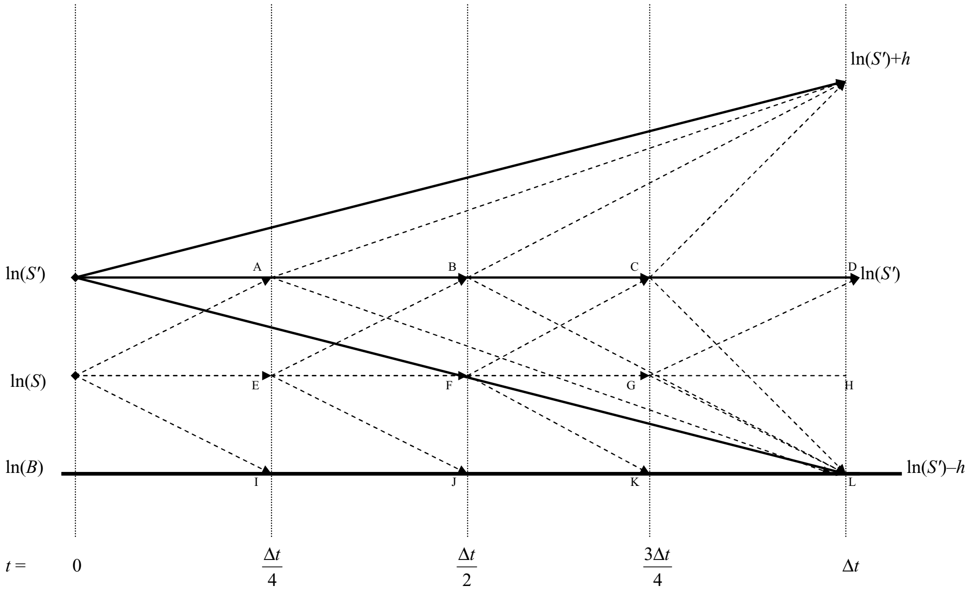


Figure I. Illustration of Geometric Brownian Motion Adaptive Mesh. The initial asset price is given by S. S' is the asset price that provides the initial price point for the course mesh. B is the lower barrier of the option. A single time step in the course mesh is given by Δt. h is the course trinomial jump size of the log asset price.

Figure I illustrates the implementation of the adaptive mesh model for a down-and-out option for the first step of a trinomial tree.[2] An option with an upper barrier would yield a horizontal reflection of this figure. Let the lower barrier level be given by B. The price of the barrier option is desired for an initial asset price S. We construct a coarse trinomial tree as if the initial asset price is $\ln(S') \equiv \ln(S) + h/2$, where h is determined by applying the Ritchken procedure to S'. In Figure I this coarse mesh is given by the three branching solid lines. The dotted lines represent the finer mesh. The spacing as illustrated reveals a time step for this finer mesh at one-fourth the spacing of the coarse mesh.

The finer mesh consists of the points labeled A through L, as well as the time 0 point at $\ln(S)$. The purpose of this finer mesh is to capture the information in the coarse mesh, augment it with the information contained in the finer mesh, and use the two layers together to determine the time 0 value of the option when the asset price is S.

Once the terminal nodes of the coarse lattice step have been determined, it is then possible to begin working backward through the fine mesh. To link the fine mesh to the coarse mesh, we first compute nodes A, B, and C. The value of

[2] Figure I is a subset of the information presented in Figure VI of Figlewski and Gao (1999a).

the option at each of these points may be determined by drawing on information available at the terminal coarse nodes. The probabilities for the up, down, and middle ticks of the trinomial steps emanating from these three points are found by solving equations analogous to those presented in equations (2) (see Figlewski and Gao 1999a). Option values at these points are then determined by using risk-neutral pricing—the option values are known for the coarse lattice. Option values for points I, J, and K, along the barrier, are of course 0.

Once the information is known for points A, B, C, D, H, I, J, K, and L, it is possible to determine the option values at points E, F, and G, and ultimately the value of our option at time 0, asset price S. Subsequent time steps proceed analogously. This method can dramatically decrease the computational time necessary to price barrier options accurately, particularly when the asset price is close to the barrier.

The adaptive mesh described here is a single layer of fine mesh. Figlewski and Gao (1999a) refer to this as AMM-1. It is possible to attach successive fine layers to the model by repeating the process described here. In this way, it is possible to build very accurate models based on relatively coarse trinomial trees.

III. The Jump-Diffusion Process and the Pricing of Path-Dependent Options

Although GBM provides a well-understood environment in which we can examine the effectiveness of the adaptive mesh model, it is primarily when the underlying asset follows more complicated dynamics that the methodology should prove most useful. Figlewski and Gao (1999a) provide no such extension. Asset prices have been shown to exhibit both skewness and excess kurtosis, characteristics that cannot be captured by the lognormal asset return model of GBM.

Several empirically viable alternative models have been proposed. Bakshi, Cao, and Chen (1997) review and compare these models in the context of equity index options—one of many underlying assets in which barrier option prices may need to be determined. One such model that has had success in describing the fluctuations in asset prices is the jump-diffusion model. Consider the following risk-neutralized specification of the dynamics of the underlying asset:[3]

$$\frac{dS}{S} = (r - d - \lambda\bar{k})\,dt + \sigma dz + kdq. \tag{6}$$

This specification of the underlying process allows a continuous diffusion during the "typical" evolution of the asset price, with the occasional "jump" in the asset price due to the arrival of significant information concerning the asset price. The smooth

[3]We use notation consistent with Bates (1991) and Hilliard and Schwartz (2005).

process is driven by standard Brownian motion, $dz,$ magnified by the volatility parameter σ. dS and dt denote infinitesimal changes in the asset price and time, respectively. r is the riskless rate of interest, and d remains the proportional dividend yield continuously paid by the underlying asset, as in equation (1). k is a random variable that determines the magnitude of a jump, an event that only occurs when $dq = 1$. The arrival of jumps is governed by a Poisson process, with intensity parameter λ; $dq = 0$ at all other times. $\ln \ (1 + k) \sim N(\gamma', \delta^2)$, and $\gamma' \equiv \gamma - .5\delta^2$. $E(k) \equiv \bar{k}e^{\gamma} - 1$. Merton (1976) provides the value of a European option when the underlying asset follows this process.

Hilliard and Schwartz (2005) develop a bivariate binomial tree for this jump-diffusion process. In one dimension they allow for the smooth diffusion, similar to the Cox, Ross, and Rubenstein (1979) model (hereafter denoted CRR). In the second dimension they model the Poisson jumps. Specifically, they note that the process in equation (6) may be written as

$$
\begin{aligned}
V_t &\equiv \ln\left(\frac{S_t}{S_o}\right) \equiv X_t + Y_t \\
X_t &\equiv \left(r - d - \lambda\bar{k} - \frac{\sigma^2}{2}\right) t + \sigma z(t), \\
Y_t &\equiv \sum_{i=0}^{n(t)} \ln(1 + k_i),
\end{aligned}
\tag{7}
$$

where $z(t)$ is $\int_0^t dz$ and $n(t)$ is the Poisson distributed number of jumps from 0 to t.

Hilliard and Schwartz (2005) model the smooth component as a binomial process. However, the convergence of this binomial model exhibits the sawtooth convergence for the pricing of barrier options similar to that reported in Boyle and Lau (1994). We present a trinomial version that provides greater flexibility than the binomial and refer readers to Hilliard and Schwartz for details of the original model. Following that model, we allow the number of possible jumps from a given point in the lattice, which captures the jump component of equation (7), to be chosen by the modeler. Specifically, we define an “M-node” grid, where M $= 2m + 1$. m is both the number of possible “up” jumps and the number of possible “down” jumps in the tree. Also, it is possible that the asset experiences no jumps. The magnitude of these jumps are separated by the parameter η. Therefore, a single step on the $2 \times$ M grid is defined as:

$$V^+_{t+\Delta t,j} = V_t + h + j\eta, \qquad j = 0, \pm 1, \pm 2, \ldots, \pm m \tag{8}$$

$$V^0_{t+\Delta t,j} = V_t + j\eta, \qquad j = 0, \pm 1, \pm 2, \ldots, \pm m \tag{9}$$

$$V^-_{t+\Delta t,j} = V_t - h + j\eta, \qquad j = 0, \pm 1, \pm 2, \ldots, \pm m, \tag{10}$$

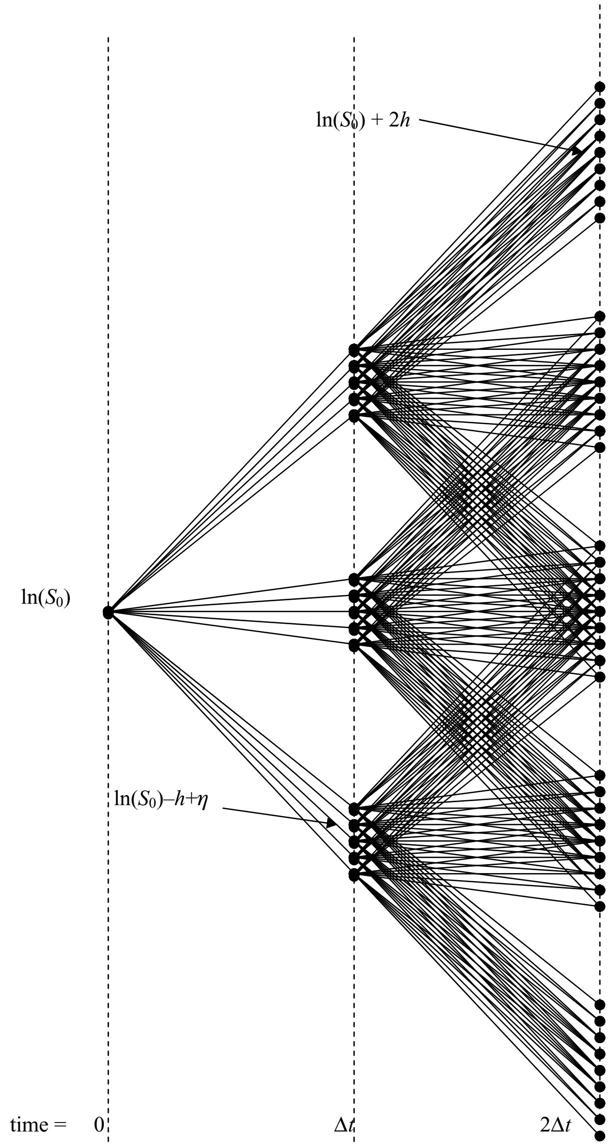


Figure II. Hillard and Schwartz Trinomial Tree in Two Dimensions. S_0 is the initial asset price. h and η are the smooth and discrete jump size of the log asset prices, respectively. A single time step in the course mesh is given by Δt.

where Δt is the length of an individual time step. Figure II provides a two-dimensional illustration for a five-node grid. With the five-node tree, each of the three "groups" at time Δt represent the smooth tick as well as the four possible discrete nonzero jumps. The distance between a smooth uptick and downtick at Δt

is given by h. Notice that although the nodes per group grows at each time step, the nodes do recombine. Two points on this graph have been labeled for clarity. The starting value of $\ln(S_0)$ is also given as a reference point.[4] Hilliard and Schwartz (2005) choose η to satisfy

$$\eta = \alpha\sqrt{(\gamma')^2 + \delta^2}, \tag{11}$$

where α is set to 1, a value they determine provides greater accuracy for the model. We continue to use this definition for η.

To determine the option prices, we must employ backward recursion. Denote the value of a derivative at time t by $F(V_i,t)$. The backward recursion to determine this option value is given by

$$\begin{aligned} F(V_i, t) = e^{-r(\Delta t)} \sum_{j=-m}^{m} [&p_u q(j) F(V_i + h + j\eta, t + \Delta t) \\ &+ p_m q(j) F(V_i + j\eta, t + \Delta t) \\ &+ p_d q(j) F(V_i - h + j\eta, t + \Delta t)]. \end{aligned} \tag{12}$$

The $q(j)$ are the probabilities of a given jump realization (independent of the outcome of the smooth process and summing to one), and p is the probability of a given smooth tick.

We choose the smooth probabilities so that the moments of the discrete tree match those of the underlying continuous process. As we have three possible smooth steps, we match the first two moments of the underlying continuous process, as well as a restriction that the probabilities sum to one. Similar to those presented in equation (2), this leaves the following system of equations to be solved for the probabilities:

$$\begin{aligned} &E(\ln(S(t+\Delta t) - \ln(S(t)) = \left(r - d - \lambda\bar{k} - \frac{\sigma^2}{2}\right)\Delta t = p_u h + p_m 0 + p_d(-h) \\ &E([\ln(S(t+\Delta t) - \ln(S(t)]^2) = \sigma^2 \Delta t = p_u h^2 + p_m 0 + p_d(-h)^2 \\ &1 = p_u + p_m + p_d \end{aligned} \tag{13}$$

[4] Although the spacing of nodes within a group has been illustrated to be comparatively small (indicating a small η), this is not necessarily the case.

for which the solution is

$$p_u = \frac{\left(r - d - \lambda\bar{k} - \frac{\sigma^2}{2}\right)\Delta t}{2h} + \frac{\sigma^2 \Delta t}{2h^2}$$
$$p_m = 1 - \frac{\sigma^2 \Delta t}{h^2} \tag{14}$$
$$p_d = \frac{\sigma^2 \Delta t}{2h^2} - \frac{\left(r - d - \lambda\bar{k} - \frac{\sigma^2}{2}\right)\Delta t}{2h}.$$

Notice that if the arrival of jumps is eliminated (i.e., $\lambda = 0$), these probabilities reduce to the probabilities derived from the GBM case in equation (3). As we are modifying this model with the goal of pricing barrier options, we parallel the Ritchken (1995) technique for selecting h in the GBM case. As in equation (5), we eliminate the possibility of a nonnegative probability by defining a free parameter $\Lambda_J > 1$ such that

$$\Lambda_J = \frac{h^2}{\sigma^2(\Delta t)}, \tag{15}$$

again allowing this to establish the relationship between h and Δt. In our implementation of this model, we use $\Lambda_J = 3$ because of its established convergence properties. h is chosen to allow an integer number of time steps between the log asset price and the log barrier price.

The discrete jump probabilities are also given by a moment-matching condition.[5] The $q(j)$ probabilities are chosen to match the first M–1 moments of Y, or

$$\sum_{j=-m}^{m} (j\eta)^{i-1} q(j) = \mu'_{i-1} \equiv E\left[\sum_{j=0}^{n(\Delta t)} \ln(1 + k_j)\right]^{i-1}, \; i = 1, 2, \ldots, M \tag{16}$$

where μ'_{i-1} is the ith local moment of Y. $i = 1$ yields the condition that the probabilities must sum to one. The equations in (16) may be easily solved for $q(j)$.

Hilliard and Schwartz (2005) produce simulation results that indicate that the binomial model improves as jump nodes are added until we reach the seven-node specification.

For the trinomial tree, like the binomial version, each time step i has $i^*(M - 1) + 1$ nodes per group. Whereas the binomial tree has $(i + 1)$ groups

[5] Strictly speaking, Hilliard and Schwartz (2005) determine these probabilities by matching cumulants, which provide approximations for the relevant moments. The Appendix lists the relevant local cumulants.

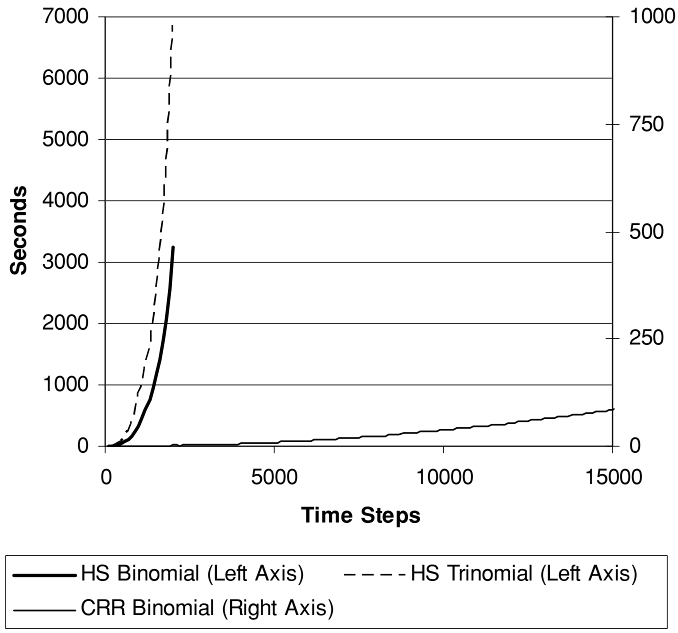


Figure III. Run Time Comparison: Various Models. Computational time for a Cox, Ross, and Rubenstein (1979) (CRR) binomial tree and seven-node Hilliard and Schwartz (2005) jump binomial and trinomial trees. Computational time is in seconds and run on a PC with and Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The program is compiled and executed in C++. The Hilliard and Schwartz times are reported on the left axis, and the CRR model is reported on the right axis.

per time step, the trinomial tree has $(1 + 2i)$ of them. In either case the number of nodes necessary to compute the option value is explosive. Figure III illustrates the computational time necessary to price a European call on seven-node Hilliard–Schwartz binomial and trinomial trees and a standard CRR binomial tree. Whereas the total number of nodes on the n-step CRR binomial tree is given by $(n+1)(n+2)/2$, the total number of nodes on the Hilliard–Schwartz binomial and trinomial trees are

$$\sum_{i=0}^{n}(1+i)(i(M-1)+1) \quad \text{and} \quad \sum_{i=0}^{n}(2i+1)(i(M-1)+1)$$

for an n-step, M-node tree, respectively. Whereas a 1,000-step CRR tree takes approximately 1 second to run, a seven-node, 1,000 step Hilliard– Schwartz binomial tree takes 336 seconds and the trinomial 872 seconds to run on the same machine.[6] This dramatic time increase is attributable to the number of nodes. The

[6]Computational time is determined on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The program is compiled and executed in C++.

Hilliard–Schwartz binomial model must compute 2,006,505,500 nodes for this tree. For the CRR model, there are 501,501 nodes. We timed the Hilliard–Schwartz model only up to 2,000 time steps (machine RAM did not permit many time steps beyond this), which took 54 minutes and 14 seconds for the binomial and 114 minutes and 28 seconds for the trinomial. The corresponding CRR model took 2 seconds.

These computational issues of the Hilliard and Schwartz model are largely irrelevant for the pricing of American options. Hilliard and Schwartz (2005) document effective convergence in 600 time steps, which we determine has a run time of just a few minutes. They report other results that indicate even 200 time steps may be sufficient. However, for the effective pricing of a barrier option a substantially greater number of time steps may be required. The Hilliard–Schwartz binomial and trinomial trees exhibit the same sawtooth convergence patterns for barrier options as their GBM counterparts. Consider an example of the Ritchken (1995) technique of choosing h so that a single smooth downtick (with no jumps) lands on the lower barrier of the down-and-out option. We assume a reasonable parameratization (using the parameters of Panel A, Table 1 of Hilliard and Schwartz 2005) of $\sigma = .2236$, $\lambda = 5$, $\gamma = 0$, and $\delta = .2236$, which imply $\gamma' = -.025$. Further assume that the asset price is 50 and the barrier is 49.75. The largest h that permits a single downtick before hitting the barrier is $h = \ln(50) - \ln(49.75) = .0050125$. If $\Lambda_J = 3$, this implies $\Delta t = .000158$, or 6,343 steps in the tree. This tree requires us to compute more than 1.02 trillion nodes. We estimate that this model will take almost three days to compute, even if the necessary RAM were available. Similar results are easily demonstrable for up-and-out options.

The adaptive mesh model provides significant run-time improvements relative to the standard trinomial. As we have seen, the Hilliard–Schwartz model increases in computational time even faster than the standard models. In the next section we implement an extension of the standard adaptive mesh into the Hilliard–Schwartz trinomial model, and demonstrate that the computational benefit of such an extension is significant, rendering feasible a range of parameterizations for a variety of barrier options that were previously infeasible.

IV. Extension of the Adaptive Mesh to the Jump-Diffusion Trinomial

Extension of the adaptive mesh model into the jump-diffusion trinomial requires a generalization of the results from Section II. Our goal in this more complex setting remains the grafting of a fine mesh tree onto a coarse tree that is able to satisfy the Ritchken (1995) requirement of having an integer number of time steps until the barrier with a tractable overall node count.

Figure IV illustrates the jump trinomial adaptive mesh constructed for a down-and-out option, using an illustrative shorthand for the number of nodes at each smooth tick grouping. For example, because the fine mesh in the figure is a

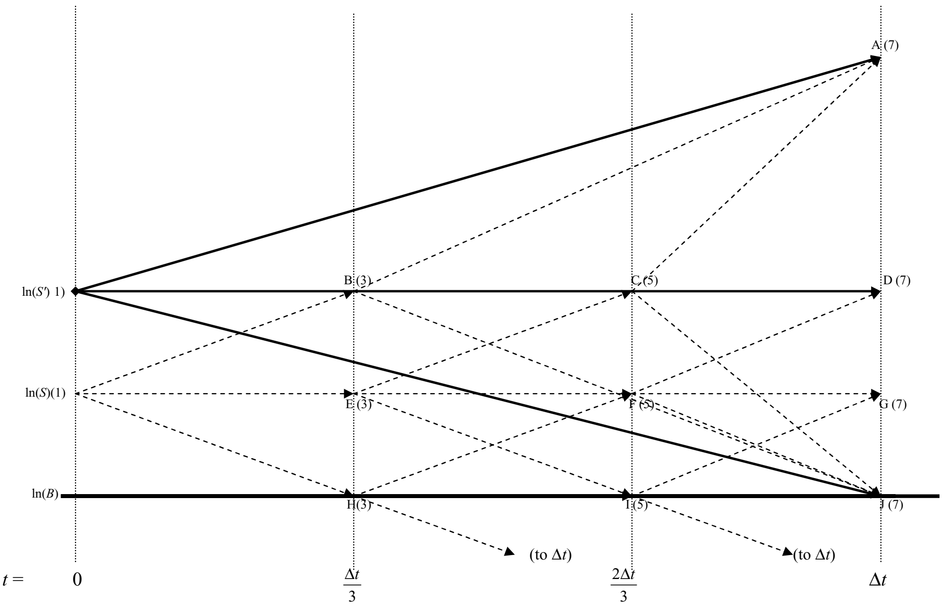


Figure IV. Jump Trinomial Adaptive Mesh. This figure illustrates the nodes in the first step of an adaptive mesh model implemented on a jump diffusion trinomial. The initial asset price is given by S. S' is the asset price that provides the initial price point for the course mesh. B is the lower barrier of the option. A single time step in the course mesh is given by Δt. h is the course trinomial jump size of the log asset price. Numbers in parentheses indicate the number of nodes in the group. The coarse mesh is assumed to be a seven-node tree, and the finer mesh is a three-node tree. Points B and C draw information from all seven nodes of the coarse lattice.

three-node mesh, the nodes of the second fine time step—C, F, and I—have parenthetical fives next to the nodes, indicating that there are five nodes in each group. Figure IV is analogous to Figure I, though there are some important differences.

The coarse tree in Figure IV is a seven-node jump trinomial, which we illustrate here near the barrier of a down-and-out option at time 0 with asset price S.[7] However, we define $h/2 \equiv \ln(S) - \ln(B)$, and construct the coarse tree as if the current asset price is S', defined to satisfy $\ln(S') = \ln(B) + h$. The concluding nodes of this single coarse step are nodes A, D, and J. The spacing of these nodes in the time dimension is determined by h and the relation defined in equation (15), assuming the lattice begins with asset price S'.

The fine tree begins with asset price S. Like the adaptive mesh in Figure I, a single coarse step is subdivided into multiple fine steps. At this point arbitrarily, each fine step in this example is $(1/3)\Delta t$. It is not desirable for this finer tree

[7]The fine mesh is placed analogously above the coarse mesh in the instance of a barrier option with an upper barrier.

to have as many jump nodes as the coarse tree. If the fine tree were a 7-node tree, the fine mesh groups at points D, G, and J would each have 19 nodes. This would not align with the number of coarse nodes at points D and J. Either new information would have to be computed or the model would not be completely specified.

For this example, the fine tree is modeled as a three-node tree. However, we first determine the value of the "anchor" groups—B and C. These groups have three and five constituent nodes, respectively, in accordance to their generation from a three-node model. However, they each draw time Δt information from groups containing seven constituent nodes. For these groups to capture the maximum possible information, we allow these anchor groups to branch into all nodes of the A, D, and J groups (in this first time step, this is seven nodes). We again have two sets of probabilities that must be determined, the *p*s that determine the movement in the "smooth" space, and the *q*(*j*)s that determine the discrete jump movements. For each node group B and C, we must solve a system of equations similar to those presented in equation (13). The only modification must be to use a time step of $\frac{2\Delta t}{3}$ for each of the nodes in group B, and a time step of $\frac{\Delta t}{3}$ for the nodes in group C. The resulting probabilities are a minor generalization of equation (14). Specifically, $y\frac{\Delta t}{3}$ substitutes in that equation for Δt, where $y = 1$ in determining group C probabilities and $y = 2$ for group B.

Similarly for each group, the *q*(*j*)s are determined by a series of equations similar to those listed in equation (17):

$$\sum_{j=-m}^{m} (j\eta)^{i-1} q(j) = \mu'_{i-1} \equiv E\left[\sum_{j=0}^{n(y\Delta t)} \ln(1+k_j)\right]^{i-1}, \quad i, \ldots, M, \tag{17}$$

where $M = 2m + 1$. Note, however, that the value of m is different for groups B and C. Although group C is defined in the standard way, group B generates more paths than would be typical for that group so as to match the seven nodes that exist at points A, D, and J. Specifically, because there are three nodes within the group, this group will be allowed to branch as if it were a five-node step. Therefore m in equation (17) for each of the nodes in group B is 2. By contrast, m for the nodes in group C is 1.

Groups H and I are determined analogously. Although the groups are drawn on the barrier, only nodes with net zero jumps fall exactly on the barrier. Those with net negative jumps are below the barrier (and thus worthless), but those with net positive jumps lie above the barrier and must be assigned a value. It is therefore necessary for some nodes in groups H and I to draw information from groups D, J, and a third group below the illustration at time Δt.

Now that the anchor groups of B, C, H, and I have been determined, we may establish the groups that comprise the finer mesh—F, E, and the starting node (we assume that group G is known, as it is determined from information generated between time Δt and $2\Delta t$). Group F draws option information from each of the seven nodes in each of the three groups D, G, and J. Group E draws on information from each of the five nodes in each of the groups C, F, and I. Lastly, the starting node draws information from each of the three nodes in each of the groups B, E, and H. Although we only use a three-node fine mesh, the information from the coarse mesh should have filtered effectively into the final solution.

The probabilities associated with the starting node and groups E and F are straightforward to calculate. In all cases, the time step is given by $\frac{\Delta t}{3}$. Furthermore, in all cases the smooth jump uptick magnitude is $h/2$. Solving equations similar to equation (13) with Δt replaced by $\frac{\Delta t}{3}$ yields the appropriate risk-neutral probabilities.

The discrete jump probabilities solve equations similar to (16):

$$\sum_{j=-m}^{m} (j\eta)^{i-1} q(j) = \mu'_{i-1} \equiv E\left[\sum_{j=0}^{n(\Delta t/3)} \ln(1+k_j)\right]^{i-1}, i = 1, 2, \ldots, M. \qquad (18)$$

We define M_2 to be the number of jump nodes in the fine mesh. In this example, $m = 1$, resulting in an $M_2 =$ three-node fine tree.

We note earlier that the finer tree must have fewer jump nodes than the coarse mesh. Specifically, it must be the case that

$$1 + (M_2 - 1)n_2 \leq M, \qquad (19)$$

where n_2 is the number of fine mesh time steps per step of the coarse mesh. In the example presented in Figure IV, $M_2 = 3$, $n_2 = 3$, and $M = 7$, causing equation (19) to hold with equality. If the left-hand side of equation (19) is greater than the right-hand side, eventually the number of nodes per group in the fine mesh will exceed the number of nodes per group in the coarse mesh, resulting either in an incomplete model or the need to generate nodes (and consequently significantly increase computer run time). It is only when equation (19) holds as an equality that the maximum amount of information is being generated within the fine mesh, relative to the coarse mesh.

Equation (19) also imposes a limit on the adaptive mesh in the Hilliard–Schwartz model that is not present in the standard trinomial. To add additional layers of mesh in the jump-diffusion world, an equation such as (19) must hold between the first and the second layers of the mesh. It is always possible to add an arbitrary number of layers of standard trinomial mesh with the probabilities chosen to match the moments of the jump diffusion process, but the convergence properties of such an approach have yet to be fully explored.

V. Empirical Results

In unreported results, we compare the pricing accuracy of the trinomial modification of the Hilliard–Schwartz model with the original binomial model and the analytic value for European options. For a reasonable set of parameterizations, we confirm that both the binomial and trinomial models, even for 200 time steps, are able to accurately price the European put options.

Tables 1a and 1b present values and run times for the Hilliard–Schwartz trinomial, the Hilliard–Schwartz trinomial model with the Ritchken (1995) modification of the asset price landing on the barrier after exactly one downtick or uptick (and thereby setting the number of time steps), and our AMM implementation of the trinomial model with the same Ritchken modification. We use each model to evaluate down-and-out calls and puts as well as up-and-out calls and puts. We use a base case of parameters and price down-and-out options with a barrier of $B = 100$ and up-and-out options with a barrier of $B = 110$. Our base case here is $X = 105$, $\sigma = .1$, $\lambda = 3$, $\gamma = 0$, $\delta = .05$, $r = .05$, and $T = 1$. We provide simulated values for comparison.

Panel A of Tables 1a and 1b presents the results for this base case of parameters as the stock price approaches the barrier. Upon initial inspection, the significant inaccuracy in both the Ritchken and AMM-1 modifications for large asset price distances from the barrier value may seem puzzling. However, because the number of time steps in both models is inversely related to the size of this spread, this is to be expected. Because of the large spread, the number of time steps is too small for any lattice model to be accurate. As the table shows, the accuracy of both models improves significantly as the spread decreases. Both of these models are intended to be used when the asset price is close to the barrier, and as the tables show, they are accurate in this range.

However, as the asset prices approach the barrier, the trinomial model with a fixed number of time steps yields pricing solutions with potentially great error. For example, when pricing an up-and-out call when $S = 109.5$, the 600-step trinomial tree gives a value for the option (*100) of \$.464, a difference of roughly \$.182 from the simulated value of about \$.146, resulting in an error of about 65%.[8] The Ritchken modification of the trinomial model provides a much more accurate solution, coupled with a run time of about 56 minutes and 43 seconds on the test computer. By comparison, the AMM model yields a similarly accurate solution with an error within the confidence bounds and a run time of 48 seconds—about 1.41% the run time of the Ritchken model. The run-time advantage of the AMM procedure is more pronounced for asset prices closer to the barrier. With an asset

[8]The 600-step binomial tree has a run time of 213 seconds on our test computer.

TABLE 1a. Adaptive Mesh Model (AMM) Performance, the Effect of Stock Price Approaching the Barrier of Down-and-Out Options.

Panel A. Base Case Option Values

$X = 105, B = 100, \sigma = .1, \lambda = 3, \gamma = 0, \delta = .1, r = .05, T = 1$

	Simulation		HS Trinomial	HS Ritchken Trinomial		AMM Ritchken Trinomial	
Stock Price	Value	+/– 95% Bounds	600-Step Value	Value (Steps)	Run time	Value (Steps)	Run time
			Down-and-Out Call				
101	2.109	(2.097, 2.121)	2.816	2.108 (303)	28	2.150 (75)	< 1
100.75	1.621	(1.611, 1.631)	2.766	1.620 (537)	153	1.652 (134)	3
100.5	1.126	(1.117, 1.126)	1.519	1.106 (1,206)	1,786	1.118 (301)	28
100.25	.593	(.587, .599)	1.498	— (4,812)	—	.570 (1,203)	1,764
			Down-and-Out Put (Option Values $*10^2$)				
101	.694	(.676, .712)	1.227	.727 (303)	28	.792 (75)	< 1
100.75	.519	(.504, .534)	1.221	.585 (537)	153	.671 (134)	3
100.5	.367	(.354, .380)	.577	.375 (1,206)	1,786	.385 (301)	28
100.25	.192	(.182, .201)	.663	— (4,812)	—	.193 (1,203)	1,764

Panel B. Base Case Delta Values

	Simulation		HS Trinomial	HS Ritchken Trinomial	AMM Ritchken Trinomial
Stock Price	Delta Value	+/– 95% Bounds	Delta Value	Delta Value	Delta Value
			Down-and-Out Call		
101	1.849	(1.824, 1.873)	1.728	1.910	1.901
100.75	1.962	(1.936, 1.987)	1.701	2.005	2.007
100.5	2.075	(2.048, 2.101)	1.998	2.101	2.099
100.25	2.186	(2.158, 2.213)	1.975	—	2.208
			Down-and-Out Put (Delta Values $*10^2$)		
101	.604	(.553, .655)	.761	.662	.690
100.75	.632	(.584, .680)	.754	.730	.825
100.5	.657	(.611, .703)	.766	.718	.722
100.25	.741	(.697, .785)	.883	—	.748

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard–Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

TABLE 1b. Adaptive Mesh Model (AMM) Performance, the Effect of Stock Price Approaching the Barrier of Up-and-Out Options.

Panel A. Base Case Option Values

$X = 105, B = 110, \sigma = .1, \lambda = 3, \gamma = 0, \delta = .1, r = .05, T = 1$

	Simulation		HS Trinomial	HS Ritchken Trinomial		AMM Ritchken Trinomial	
Stock Price	Value	+/– 95% Bounds	600-Step Value	Value (Steps)	Run time	Value (Steps)	Run time
Up-and-Out Call (Option Values $*10^2$)							
109	.544	(.527, .600)	1.051	.566 (359)	48	.659 (89)	1
109.25	.403	(.388, .417)	.475	.435 (640)	270	.496 (160)	4
109.5	.282	(.271, .294)	.464	.277 (1,445)	3,403	.289 (361)	48
109.75	.146	(.138, .154)	.529	— (5,794)	—	.150 (1,448)	3454
Up-and-Out Put							
109	.763	(.758, .769)	1.117	.759 (359)	48	.783 (89)	1
109.25	.579	(.574, .584)	.592	.574 (640)	270	.588 (160)	4
109.5	.373	(.369, .377)	.582	.385 (1,445)	3,403	.391 (361)	48
109.75	.205	(.202, .208)	.575	— (5,794)	—	.196 (1,448)	3454

Panel B. Base Case Delta Values

	Simulation		HS Trinomial	HS Ritchken Trinomial	AMM Ritchken Trinomial
Stock Price	Delta Value	+/– 95% bounds	Delta Value	Delta Value	Delta Value
Up-and-Out Call (Delta Values $*10^2$)					
109	−.504	(−.550, −.459)	−.658	−.557	−.614
109.25	−.555	(−.599, −.510)	−.600	−.570	−.624
109.5	−.473	(−.511, −.434)	−.590	−.550	−.564
109.75	−.534	(−.572, −.497)	−.671	—	−.589
Up-and-Out Put					
109	−.727	(−.740, −.715)	−.684	−.734	−.734
109.25	−.727	(−.740, −.715)	−.744	−.745	−.746
109.5	−.723	(−.736, −.710)	−.730	−.757	−.757
109.75	−.762	(−.750, −.776)	−.720	—	−.770

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard–Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

price of \$109.75, the Ritchken modified trinomial requires 5,794 time steps and may not be computed on our test computer. By comparison, the AMM modification that yields results within the simulation confidence bounds can be computed on the test machine in about one hour.

Panel B of Tables 1a and 1b compares the hedging parameters of the three lattice models. The fixed-step trinomial continues to fare poorly. The delta of the Ritchken trinomial model and the AMM trinomial provide similar results and good accuracy in matching the delta of the options.[9]

So what drives the run-time improvements? For the Ritchken technique, the value of h is given by $\ln(S) - \ln(B)$. This, combined with equation (15) and recognizing that $\Delta t = \frac{T}{N}$ when N is the number of time steps in the tree, yields

$$\ln(S) - \ln(B) = \left(\sigma\sqrt{\frac{T}{N}}\right)\sqrt{\Lambda_J}. \tag{20}$$

Because Λ_J is fixed, as S approaches B, N must get larger, resulting in longer run times.

As equation (20) makes clear, for a given stock price and barrier level (and thus a given, fixed h) a doubling of the smooth process volatility necessitates a quadrupling of time steps. A doubling of the expiration date of the option necessitates a doubling of the number of time steps.

The Ritchken technique is also employed for the AMM. The creation of a coarse mesh from a higher initial stock price, and then grafting a finer mesh onto the lattice simply reduces the number of necessary time steps—it does not alter the underlying relation. Specifically, by building a coarse mesh using an h that is twice as large as would be the case under the standard Ritchken procedure, the relation becomes

$$2(\ln(S) - \ln(B)) = \left(\sigma\sqrt{\frac{T}{N}}\right)\sqrt{\Lambda_J}, \tag{21}$$

---

[9]The deltas of both the Ritchken modified trinomial and the AMM trinomial are consistently reliable. For example, in unreported results, we find that of the 40 down-and-out calls for which we have simulated prices and deltas (and for which the AMM had more than four time steps), the average difference between the trinomial and AMM deltas is .26%, with a maximum of 1.9%. The average AMM deviation from the simulated delta value is –1.7%, and the worst performance was a deviation of –8.2%. Consistently, as the stock approached the barrier (and therefore time steps were added to both models), the deltas converged to the simulated values. Results were similar across instruments. Because the models performed well and so similarly, this similarity is noted and excluded from the ensuing analysis.

or

$$\ln(S) - \ln(B) = \left(\sigma\sqrt{\frac{T}{4N}}\right)\sqrt{\Lambda_J}. \tag{22}$$

Therefore, the AMM-1 model has only one-fourth the coarse mesh time steps of the standard trinomial. This is true in all cases. Given the explosive computational cost of increasing time steps in the Hilliard–Schwartz model, this becomes particularly important when the $\ln(S)$ is close to $\ln(B)$, when σ is high, or when time to expiration T is large.[10]

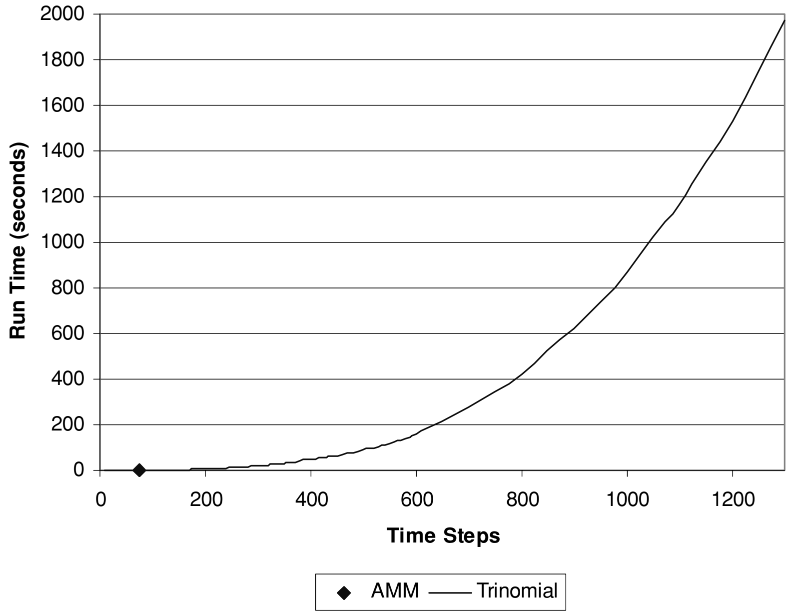
The computational benefit of the AMM in the case of a down-and-out call is illustrated in Figure V. Panel A shows the run times for the jump trinomial model for a range of time steps and for the AMM modification. Panel B gives the values of the trinomial model over the same range of time steps, the Ritchken modification with both one and two steps to the barrier, and the AMM modification. Also, Panel B gives the true option value. The set of parameters are arbitrarily chosen here to be $X = 105$, $B = 100$, $\lambda = 3$, $\gamma = 0$, $\delta = .1$, $r = .05$, $\sigma = .05$, $T = 1$, and $S = 100.5$. The AMM and Ritchken modifications are shown at only one time step each because spread between the asset price and the barrier determines the number of time steps until expiration.

In Panel B of Figure V we see the sawtooth convergence toward the true option price. For the trinomial method to achieve an accurate result, the model must have 301 time steps. Because of the high computational cost of additional time steps, the AMM model, with only 75 required time steps, significantly reduces run time. Panel A illustrates these computational benefits as the run time is reduced from 28 seconds for a one-step Ritchken modification to less than one second for the AMM modification.

The run time improvement of the AMM implementation is most significant for high values of the volatility of the continuous process. Tables 2a and 2b illustrate this using our base case of parameters but allow σ to take values of .05, .1, and .15. The 600-step trinomial implementation exhibits unreliable pricing whenever the asset price is close to the barrier, and the Ritchken modified trinomial continues to perform well. However, the higher volatility necessitates a higher number of time steps and therefore a longer run time. For all instruments, notice the quadrupling of the time steps when the volatility is doubled from .05 to .1, for both the Ritchken trinomial and the AMM models. This is in agreement with equations (20)–(22). Because of the rapid increase in run time, use of the Ritchken trinomial becomes impractical when the stock price is even farther from the barrier than those illustrated

[10]Note that these relations hold in the GBM case exactly as they do in the jump-diffusion case discussed here.

Panel A



Panel B

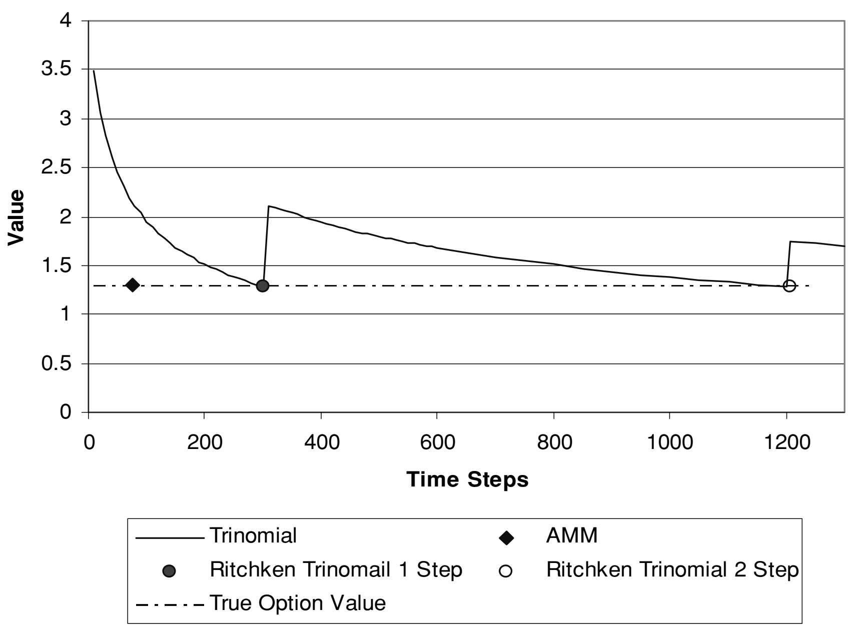


Figure V. Run Time and Value Comparison: Jump Trinomial. Reported are values for European down-and-out call options where $S = 110$, $X = 105$, $\sigma = .05$, $\lambda = 3$, $\gamma = 0$, $\delta = .05$, $r = .05$, and $T = 1$. Computational time is in seconds and run on a PC with and Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The program is compiled and executed in C++.

TABLE 2a. Adaptive Mesh Model (AMM) Performance, the Effect of Volatility, Down-and-Out Options.

Stock Price	Volatility $\sigma =$	Simulation Value	Simulation +/– 95% Bounds	HS Trinomial 600-Step Value	HS Ritchken Trinomial Value (Steps)	HS Ritchken Trinomial Run Time	AMM Ritchken Trinomial Value (Steps)	AMM Ritchken Trinomial Run Time
Panel A. Base Case Option Values								
$X = 105, B = 100, \sigma = .1, \lambda = 3, \gamma = 0, \delta = .1, r = .05, T = 1$								
Down-and-Out Call								
100.75	.05	2.758	(2.746, 2.770)	3.618	2.841 (134)	2	2.929 (33)	<1
	.1	1.621	(1.611, 1.631)	2.766	1.621 (537)	153	1.652 (134)	2
	.15	1.284	(1.274, 1.294)	1.739	1.259 (1,209)	1,765	1.277 (302)	28
100.5	.05	1.985	(1.974, 1.995)	2.669	2.024 (301)	27	2.069 (75)	<1
	.1	1.126	(1.117, 1.135)	1.519	1.106 (1,206)	1,786	1.118 (301)	28
	.15	.869	(.860, .877)	1.713	– (2,713)	–	.856 (678)	310
100.25	.05	1.086	(1.078, 1.094)	1.484	1.087 (1,203)	1,756	1.099 (300)	28
	.1	.593	(.587, .600)	1.498	— (4,812)	—	.5701 (1,203)	1,764
	.15	.455	(.449, .462)	1.681	— (10,827)	—	— (2,706)	—
Panel B. Base Case Delta Values								
Down-and-Out Put (Option Values $*10^2$)								
100.75	.05	2.097	(2.066, 2.129)	3.345	2.313 (134)	2	2.655 (33)	<1
	.1	.519	(.504, .534)	1.221	.585 (537)	153	.671 (134)	3
	.15	.224	(.214, .234)	.396	.238 (1,209)	1,832	.270 (302)	30
100.5	.05	1.463	(1.436, 1.488)	2.417	1.564 (301)	28	1.547 (75)	<1
	.1	.367	(.354, .380)	.577	.375 (1,206)	1,786	.385 (301)	28
	.15	.156	(.148, .165)	.411	— (2,713)	—	.160 (678)	329
100.25	.05	.807	(.787, .827)	1.238	.833 (1,203)	1,772	.836 (300)	29
	.1	.192	(.182, .201)	.663	— (4,812)	—	.193 (1,203)	1,764
	.15	.072	(.066, .077)	.398	— (10,827)	—	—	(2,706)

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard-Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

TABLE 2b. Adaptive Mesh Model (AMM) Performance, the Effect of Volatility, Up-and-Out Options.

	Volatility	Simulation		HS Trinomial	HS Ritchken Trinomial		AMM Ritchken Trinomial	
Stock Price	$\sigma =$	Value	+/– 95% Bounds	600-Step Value	Value (Steps)	Run Time	Value (Steps)	Run Time
Panel A. Base Case Option Values								
$X = 105, B = 110, \sigma = .1, \lambda = 3, \gamma = 0, \delta = .1, r = .05, T = 1$								
Up-and-Out Call (Option Values $*10^2$)								
109.25	.05	1.279	(1.254, 1.304)	1.403	1.331 (160)	4	1.541 (40)	<1
	.1	.403	(.403, .417)	.475	.435 (359)	270	.496 (160)	4
	.15	.232	(.221, .242)	.363	.186 (1,445)	3,403	.210 (360)	50
109.5	.05	.856	(.835, .877)	1.499	.864 (361)	49	.894 (90)	1
	.1	.282	(.282, .294)	.464	.277 (1,445)	3,403	.289 (361)	48
	.15	.150	(.142, .158)	.312	— (3,252)			
109.75	.05	.436	(.422, .451)	.705	.452 (1,448)	3,496	.484 (362)	51
	.1	.146	(.138, .154)	.529	— (5,794)	—	.150 (1,448)	3,454
	.15	.080	(.075, .087)	.356	— (13,038)	—	— (3,259)	—
Panel B. Base Case Delta Values								
Up-and-Out Put								
109.25	.05	.684	(.679, .689)	.699	.681 (160)	4	.723 (40)	<1
	.1	.579	(.574, .584)	.592	.574 (640)	270	.588 (160)	4
	.15	.550	(.545, .555)	.830	.548 (1442)	3,080	.556 (360)	48
109.5	.05	.474	(.469, .478)	.688	.460 (361)	49	.480 (90)	1
	.1	.373	(.369, .377)	.582	.385 (1,445)	3,403	.391 (361)	48
	.15	.376	(.372, .380)	.811	— (3,252)	—	.369 (813)	547
109.75	.05	.244	(.241, .247)	.357	.234 (1,448)	3,496	.239 (362)	48
	.1	.205	(.202, .208)	.575	— (5,794)	—	.196 (1,448)	3,454
	.15	.195	(.192, .198)	.803	— (13,038)	—	— (3,259)	—

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard–Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

TABLE 3a. Adaptive Mesh Model (AMM) Performance, the Effect of Time to Expiration, Down-and-Out Options.

	Time	Simulation		HS Trinomial	HS Ritchken Trinomial		AMM Ritchken Trinomial	
Stock Price	$T =$	Value	+/– 95% Bounds	600-Step Value	Value (Steps)	Run Time	Value (Steps)	Run Time
Panel A. Base Case Option Values								
				$X = 105$, $B = 100$, $\sigma = .1$, $\lambda = 3$, $\gamma = 0$, $\delta = .1$, $r = .05$, $T = 1$				
				Down-and-Out Call				
100.75	.1	.423	(.420, .426)	.478	.422 (53)	<1	.434 (13)	<1
	1	1.621	(1.611, 1.631)	2.766	1.620 (537)	153	1.652 (134)	3
	2	2.081	(2.066, 2.096)	2.673	2.072 (1,074)	1234	2.118 (268)	20
100.5	.1	.290	(.287, .292)	.367	.291 (120)	2	.295 (30)	<1
	1	1.126	(1.117, 1.126)	1.519	1.106 (1,206)	1,786	1.118 (301)	28
	2	1.442	(1.430, 1.454)	2.649	— (2,412)	—	1.428 (603)	218
100.25	.1	.152	(.150, 1.53)	.251	.151 (481)	110	.152 (120)	2
	1	.593	(.587, .599)	1.498	— (4,812)	—	.570 (1,203)	1,764
	2	.770	(.761, .779)	2.625	— (9,624)	—	— (2,406)	—
Panel B. Base Case Delta Values								
				Down-and-Out Put (Option Values $*10^2$)				
100.75	.1	22.843	(22.738, 22.948)	29.840	23.289 (53)	<1	22.141 (13)	<1
	1	.519	(.504, .534)	1.221	.585 (537)	153	.671 (134)	3
	2	.170	(.161, .179)	.248	.194 (1,074)	1274	.226 (268)	20
100.5	.1	15.553	(15.465, 15.640)	23.205	16.004 (120)	2	15.601 (30)	<1
	1	.367	(.354, .380)	.577	.375 (1,206)	1,786	.385 (301)	28
	2	.117	(.110, .124)	.287	— (2,412)	—	.124 (603)	234
100.25	.1	8.068	(8.004, 8.132)	15.979	8.207 (481)	114	8.164 (120)	2
	1	.192	(.182, .201)	.663	— (4,812)	—	.193 (1,203)	1,764
	2	.059	(.053, .065)	.328	— (9,264)	—	— (2,406)	—

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard–Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

TABLE 3b. Adaptive Mesh Model (AMM) Performance, the Effect of Time to Expiration, Up-and-Out Options.

	Time	Simulation		HS Trinomial	HS Ritchken Trinomial		AMM Ritchken Trinomial	
Stock Price	$T =$	Value	+/– 95% Bounds	600-Step Value	Value (Steps)	Stock Price	$T =$	Value
Panel A. Base Case Option Values								
				$X = 105, B = 110, \sigma = .1, \lambda = 3, \gamma = 0, \delta = .1, r = .05, T = 1$				
				Up-and-Out Call (Option Values $*10^2$)				
109.25	.1	16.986	(16.895, 17.078)	25.168	17.145 (64)	<1	16.393 (16)	<1
	1	.403	(.403, .417)	.475	.435 (359)	270	.496 (160)	4
	2	.131	(.123, .138)	.211	.145 (1,281)	2154	.168 (320)	35
109.5	.1	11.354	(11.278, 11.430)	19.008	11.529 (144)	3	11.227 (36)	<1
	1	.282	(.282, .294)	.464	.277 (1,445)	3403	.289 (361)	48
	2	.090	(.083, .097)	.243	— (2,890)	—	.0930 (722)	394
109.75	.1	5.684	(5.629, 5.738)	12.706	5.771 (579)	200	5.739 (144)	3
	1	.146	(.138, .154)	.529	—	—	.150 (1,448)	3,354
	2	.044	(.040, .049)	.276	— (11,589)	—	— (2,897)	—
Panel B. Base Case Delta Values								
				Up-and-Out Put				
109.25	.1	.213	(.210, .216)	.370	.302 (64)	<1	.311 (16)	<1
	1	.579	(.574, .584)	.592	.574 (640)	270	.588 (160)	4
	2	.525	(.519, .530)	.738	.520 (1281)	2110	.534 (320)	33
109.5	.1	.207	(.205, .209)	.283	.205 (144)	3	.208 (36)	<1
	1	.373	(.369, .377)	.582	.385 (1,445)	3403	.391 (361)	48
	2	.357	(.352, .361)	.732	— (2,890)	—	.354 (722)	380
109.75	.1	.106	(.104, .108)	.193	.105 (579)	200	.105 (144)	3
	1	.205	(.202, .208)	.575	— (5,794)	—	.196 (1,448)	3,354
	2	.195	(.193, .198)	.726	— (11,589)	—	— (2,897)	—

Note: Simulated values determined by 2,000,000 simulations, with time discretized to 256,000 pieces. HS Ritchken Trinomial is the trinomial version of the Hilliard–Schwartz model implementing the Ritchken method of locating the (smooth) downtick directly on the barrier, then choosing the number of time steps for the model. The AMM Trinomial model employs this technique for the coarse mesh. Time steps reported in the AMM model are coarse time steps. Run times (reported in seconds) are found using C++ compiled code on a PC with an Intel Pentium 4 chip with 3.4 GHz and 1 GB of RAM. The simulations took approximately three days to run. The 600-step trinomial has a run time of 213 seconds on our test computer. The dash indicates that insufficient RAM exists on the test machine to compute the option value.

in Tables 1a and 1b. In the $S = 100.5$, $\sigma = .15$ case for down-and-out options, the Ritchken model again can not be computed. By comparison, the AMM model takes 5 minutes and 10 seconds, and exhibits an error of only 1.5% for down-and-out calls and 2.5% for down-and-out puts. For the $S = 100.25$, $\sigma = .15$ case, the AMM model also exceeded the memory capacity of the computer. However, this may be overcome with a second layer of adaptive mesh. Conversely, for low volatilities, most notably at high stock prices, the AMM model performs poorly because the resultant number of time steps is too low to merit accuracy.

Tables 3a and 3b illustrate the effect of time to expiration on the run times and accuracies of the Ritchken trinomial and AMM models. As expected from an examination of equations (20)–(22), the number of time steps in either of these models increases linearly as T increases, although the AMM model has only 25% the number of time steps as the Ritchken trinomial model. For very long-dated options, such as the $T = 2$ examples demonstrated in Tables 3a and 3b, the models may reach a point at which they cannot be computed. The Ritchken model arrives at this point well before the AMM model.

The remaining parameters of the model do not affect the number of time steps in the Ritchken trinomial or AMM model. However, their magnitudes may affect the accuracy of the respective models. In unreported results, we examine the models for various jump volatilities and strike prices. As expected, the performance of the Ritchken trinomial and AMM models are comparable.[11] The 600-step trinomial tree continues to do a poor job of pricing these options.

Also unreported, we implement the AMM model to price American options when the underlying stock follows the jump-diffusion process, in a manner similar to that reported in Figlewski and Gao (1999a). We find the AMM model yields convergence patterns almost identical to the Hilliard and Schwartz (2005) results without the AMM modification.

VI. Conclusion

We present an extension of the Hilliard and Schwartz (2005) binomial bivariate pricing algorithm for jump-diffusion processes. The explosive nature of the algorithm's computational cost as time steps are added to the tree can render the model infeasible for determining the price of certain barrier options. To address this difficulty we extend this model to a trinomial framework and apply a modification of the adaptive mesh techniques presented in Figlewski and Gao (1999a).

We find a significant decline in computational time—for example, our new model easily provides 60-fold decreases in run times to obtain accurate prices for

[11] These results are available upon request from the contact author.

close-to-the-barrier knock-out options. Furthermore, we examine the benefits of this adaptive mesh modification for various parameterizations and types of barrier options relative to the standard Ritchken (1995) modification of trinomial tree jump size. We find that the adaptive mesh modification is particularly valuable when pricing options with high volatility with a long time to expiration, or a stock price very close to the barrier. There are limits to the benefits, however, as we demonstrate that the same mechanism that causes explosive run times in the Ritchken modification of the Hilliard–Schwartz model also affects the adaptive mesh implementation. The adaptive mesh implementation simply expands the computationally feasible parameter set in the dimensions of time to expiration, stock price, and (smooth) volatility.

Appendix

The following are the first 12 cumulants for the jump process under Poisson compounding:

$$
\begin{aligned}
\kappa_1 &= \lambda\Delta(\gamma') \\
\kappa_2 &= \lambda\Delta((\gamma')^2 + \delta^2) \\
\kappa_3 &= \lambda\Delta((\gamma')^3 + 3(\gamma')\delta^2) \\
\kappa_4 &= \lambda\Delta((\gamma')^4 + 6\delta^2(\gamma')^2 + 3\delta^4) \\
\kappa_5 &= \lambda\Delta((\gamma')^5 + 10\delta^2(\gamma')^3 + 15\delta^4(\gamma')) \\
\kappa_6 &= \lambda\Delta((\gamma')^6 + 15\delta^2(\gamma')^4 + 45\delta^4(\gamma')^2 + 15\delta^6) \\
\kappa_7 &= \lambda\Delta((\gamma')^7 + 21\delta^2(\gamma')^5 + 105\delta^4(\gamma')^3 + 105\delta^6(\gamma')) \\
\kappa_8 &= \lambda\Delta((\gamma')^8 + 28\delta^2(\gamma')^6 + 210\delta^4(\gamma')^4 + 420\delta^6(\gamma')^2 + 105\delta^8) \\
\kappa_9 &= \lambda\Delta((\gamma')^9 + 36\delta^2(\gamma')^7 + 378\delta^4(\gamma')^5 + 1260\delta^6(\gamma')^3 + 945\delta^8(\gamma')) \\
\kappa_{10} &= \lambda\Delta((\gamma')^{10} + 45\delta^2(\gamma')^8 + 630\delta^4(\gamma')^6 + 3150\delta^6(\gamma')^4 + 4725\delta^8(\gamma')^2 \\
&\quad + 945\delta^{10}) \\
\kappa_{11} &= \lambda\Delta((\gamma')^{11} + 55\delta^2(\gamma')^9 + 990\delta^4(\gamma')^7 + 6930\delta^6(\gamma')^5 \\
&\quad + 17325\delta^8(\gamma')^3 + 10395\delta^{10}(\gamma')) \\
\kappa_{12} &= \lambda\Delta((\gamma')^{12} + 66\delta^2(\gamma')^{10} + 1485\delta^4(\gamma')^8 + 13860\delta^5(\gamma')^6 + 51975\delta^8(\gamma')^4 \\
&\quad + 62370\delta^{10}(\gamma')^2 + 10395\delta^{12})
\end{aligned}
$$

References

Amin, K. I., 1993, Jump diffusion option valuation in discrete time, *Journal of Finance* 48, 1833–63.

Andricopoulos, A. D., M. Widdicks, P. Duck, and D. Newton, 2003, Universal option valuation using quadrature methods, *Journal of Financial Economics* 67, 447–71.

Andricopoulos, A. D., M. Widdicks, P. Duck, and D. Newton, 2007, Extending quadrature methods to value multi-asset and complex path dependent options, *Journal of Financial Economics* 83, 471–99.

Bates, D., 1991, The crash of '87: Was it expected? The evidence from options markets, *Journal of Finance* 46, 1009–44.

Bakshi, G., C. Cao, and Z. Chen, 1997, Empirical performance of alternative option pricing models, *Journal of Finance* 52, 2003–49.

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

Broadie, M. and O. Kaya, 2007, A binomial lattice method for pricing corporate debt and modeling chapter 11 proceedings, *Journal of Financial and Quantitative Analysis* 42, 279–312.

Brockman, P. and H. J. Turtle, 2003, A barrier option framework for corporate security valuation, *Journal of Financial Economics* 67, 511–29.

Cox, J., S. A. Ross, and M. Rubenstein, 1979, Option pricing: A simplified approach, *Journal of Financial Economics* 7, 229–63.

Figlewski, S. and B. Gao, 1999a, The adaptive mesh model: A new approach to efficient option pricing, *Journal of Financial Economics*, 53, 313–51.

Figlewski, S. and B. Gao, 1999b, Pricing discrete barrier options with the adaptive mesh model, *Journal of Derivatives* 6, 33–44.

Hilliard J. and A. Schwartz, 2005, Pricing European and American derivatives under a jump-diffusion process: A bivariate tree approach, *Journal of Financial and Quantitative Analysis* 40, 671–90.

Merton, R. C., 1976, Option pricing when underlying stock returns are discontinuous, *Journal of Financial Economics* 3, 125–44.

Press, J., 1967, A compound events model for security prices, *Journal of Business* 40, 317–35.

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