Volume 2 • Number 1
Fall 2012

JURF
Journal of Undergraduate
Research in Finance

Applications of Recombining Stochastic Volatility Trees

Jack J. Gang[1]
Princeton University
G12, G13, C63
Mark J. Bertus Prize Winner

Tree methods have been studied extensively for their effectiveness in pricing American options and other path dependent derivatives. However, when applied to stochastic volatility models, these lattice methods often become less viable due to the difficulty of constructing trees that exhibit recombination. In this paper, we present a recombining stochastic volatility tree method using the Hull and White (1994) trinomial tree embedded into a Cox, Ross, and Rubinstein (1979) binomial tree for stock option pricing. We will show that this approach efficiently prices American options and accurately captures the volatility smile observed in the market.

Keywords: Hull and White, binomial tree, trinomial tree, option pricing, stochastic volatility

Introduction

1.1 Problem Description

One of the first reliable valuation models for the value of an equity option was the Black-Scholes model. One of the biggest assumptions of the Black-Scholes model is that the volatility of the option's underlying asset is constant. However, this assumption quickly changed after the market crash of 1987 showed that the implied volatilities of stocks are not constant, but instead resemble "smiles" or "smirks." Academics and traders alike both realized that by using non-constant volatility models, they could capture these volatility surfaces.

[1] I am very grateful for the guidance of my adviser, Michael Coulon. Through his help, I have gained a greater understanding of this unique topic and our hours of discussion have challenged and grown me academically. I would also like to thank Princeton University as well as the Department of Operations Research and Financial Engineering for giving me opportunities that I could not have imagined four years ago. Finally, I would like to thank my family for how much they have done for me. Without them, I would not have had the opportunity to write this paper.

One of the methods they used to address this issue of non-constant volatility was to calculate options prices using the stochastic differential equation (SDE) following a standard model for geometric Brownian motion:

$$dS_t = \mu S_t dt + \sigma_t S_t dW_t, \tag{1}$$

where S_t is the stock price at time t, μ is the drift rate (expected return), $\{W_t, t \geq 0\}$ is a standard Wiener process, and $\{\sigma_t, t \geq 0\}$ is the stochastic volatility process, according to Hull [9].

However, there are difficulties with this approach. Since volatility is not directly observed, estimation of parameters for the model is not completely straightforward. In addition, because stochastic volatility creates incomplete markets, the option cannot be perfectly hedged with just the underlying asset; one instead must estimate a volatility risk premium, which causes the model to be difficult to calibrate.

Hull and White [10] suggested a particular model (that we utilize for stochastic volatility) that builds a trinomial tree to approximate the Ornstein-Uhlenbeck process for mean-reverting interest rates. We can use this same model to expand this method to mean-reverting stochastic volatility, which we can characterize by the following SDE:

$$d\sigma_t = \alpha(m - \sigma_t)dt + \beta dW_t^*, \tag{2}$$

where $\{\sigma_t, t \geq 0\}$ is the volatility at t, α is the mean reversion rate, m is the mean volatility value, $\{W_t^*, t \geq 0\}$ is a standard Brownian motion different from the W_t from equation (1), and β is the "volatility of the volatility." This one-factor stochastic volatility trinomial tree will be explained in detail in Section 2.

Our goal is to show that the combination of Hull and White's procedure for constructing trinomial trees for stochastic volatility with a simple binomial option pricing tree is not only viable due to recombination, but also applicable to various classes of options. We will describe this technique, analyze it in detail, show that the method has strong performance, and explore applications of the method.

1.2 Background

1.2.1 Stochastic Volatility

Stochastic volatility arose from a need to correct the shortcomings of the Black-Scholes model. The Black-Scholes formula states that there is a unique volatility across all options on the same underlying asset. However, empirical data shows that implied volatility is instead a function of strike price and maturity; implied volatility is often the least at-the-money (it increases as the strike changes in either direction) and increasing maturity usually decreases implied volatility [5].

The idea of non-constant volatility was realized decades ago by Mandelbrot (1963) and Fama (1965). Even Black and Scholes (1972) wrote while developing their model that "...there is evidence of nonstationarity in the variance. More work must be done to predict variances using the information available." Academia and industry practitioners alike knew that in a general sense, time-varying volatility would bring finance applications closer to empirical reality, which

would in turn provide people with better understanding of markets in order to make not only theories and models, but also trading decisions.

These stochastic volatility models enjoyed wide success in both academia and in the finance industry because of several desirable factors. First, the distribution of $log\ S_T - log\ S_t$ for SV models is not Gaussian (as in Black-Scholes), but instead exhibits fat tails consistent with empirical observations. Second, stochastic volatility models can simulate the negative correlation between Brownian motions that drive the asset price and volatility processes, which is a realistic industry occurrence often called the leverage effect. Lastly, these models can reproduce many features of an empirical volatility surface, including negative at-the-money skew and volatility smiles. As a result, stochastic volatility models are often the standard in industry for option valuation today.

1.2.2 Tree Methods

The first tree model for option valuation was the binomial option pricing model, proposed by Cox, Ross, and Rubinstein [4] in 1979. Since the development of this binomial tree method, many similar tree-based (or lattice) approaches to option valuation have been discussed in academia. The general concept of a tree model is that the method divides time between $T = 0$ and the option's expiration into *N* discrete periods. At a particular time $N = n$, a binomial tree has a finite number of outcomes at time $n + 1$ such that every possible change in the "state of the world" between the two periods is captured in branches. The tree then iterates this process on every path between $n = 0$ and $n = N$, estimating probabilities for every path. Finally, the values and probabilities are calculated backwards through the lattice until the value of the option at $T = 0$ is found.

The strengths of lattice-based option pricing techniques extend beyond the fact that one can visually analyze branches and nodes of a tree. Tree methods are particularly useful for pricing American options and path-dependent options; in fact, a few minor changes can add these layers of complexity to vanilla options whereas explicit solutions and numerical methods for pricing these types of derivatives are much more complicated. Computationally, trees are also often less time-intensive compared to simulation. However, tree methods also have weaknesses that we attempt to address in this paper. When a tree is particularly complex, recombination is often difficult or infeasible, making the backward calculation time-consuming if not impossible. In addition, trees tend to have quadratic or higher run times with respect to N, causing less discretized lattices to have large calculation times. We address the first issue by the particular SV tree used in this paper; in fact, recombination is one of its greatest strengths. We will explore the run time issue in Section 6.

1.3 Hull and White's Trinomial Tree for Interest Rates

One commonly used stochastic volatility model is the mean-reverting Ornstein-Uhlenbeck process (from equation (2)). Using this SDE, John C. Hull and Alan White developed a trinomial branching method to fit interest rate term structures. One of the more important results of the Hull & White method is that the trinomial lattice is capped in size because the drift of the stochastic process is mean reverting.

Hull and White's approach utilizes a trinomial tree that has three branches at each node representing interest rates (instead, we will use each node to represent a volatility state). The method then uses the SDE $d\sigma_t = \alpha(m - \sigma_t)dt + \beta dW_t^*$ and assumes that α and β are given parameters (see Section 3). The starting rate m, the time to maturity T, the number of periods N, and a constant y (specified by Hull and White [10]) are also inputted into the method.

Letting Δt equal $\frac{T}{N}$, Hull and White suggest that the spacing between each interest rate level in the tree is $x = \beta\sqrt{y\Delta t}$. They also suggest that $y = 3$, but we will discuss later why our method does not use this value. Let n be the period index from 1 to N and j be the index level of the interest rate value in the tree. When the tree branches out at each time step n, the mean-reversion is captured by one of three different trinomial branching types seen in Figure 1. Depending on the branching type of the particular node, the probabilities for each branch, as seen in Hull and White's paper, are also shown in Figure 1.

The recommendation by Hull and White is to switch to branching (c) in Figure 1 when

$$j \geq j_{max} : \left\lceil \frac{b}{\alpha \Delta t} \right\rceil, \tag{3}$$

In equation (3) Hull and White [10] choose[2] $b = 0.184$. Intuitively, Hull and White also recommend that the tree should switch to branching (b) in Figure 1 when

$$j \leq j_{min} \coloneqq -j_{max}. \tag{4}$$

Otherwise, the tree should be of branching type (a).

1.4 Other Stochastic Volatility Tree Methods

Even though the main strength of the H&W branching scheme for stochastic volatility trinomial trees is that the branching probabilities stay positive, when high correlation between underlying price and volatility is introduced (see Section 3.2), the probabilities in Figure 1 may be negative. Tseng et al. [15] calls this the lattice feasibility problem and aims to resolve it in his paper. To accomplish this, Tseng first showed that the maximum correlation that a trinomial lattice (or tree) can accommodate is directly implied by the configuration of the lattice. By adjusting the size of the grids in the tree, Tseng can create a tree that is feasible for a particular correlation range. In addition, Tseng showed that $4/\sqrt{35} \approx 0.676$ is the maximum correlation value that a feasible lattice can accommodate. However, one of the main consequences of this method (and one that our technique overcomes) is that the lattice is non-recombining.

Beliaeva [1] covered many stochastic volatility tree techniques in her thesis from 2006. Just as Tseng noted above, so too did Beliaeva note problems with negative probabilities when correlation is introduced between the stock price process and the volatility process. However,

[2] We performed further analysis and found that the value of b does not have a significant impact on the method. Minimizing b does decrease the computational time without cost to accuracy, but since b is already near the minimum value (see Section 2.1.3), we kept it at H&W's suggested value.

unlike Tseng, who used a "jump" parameter to solve this problem, Beliaeva uses a two dimensional orthogonal transform for the stochastic volatility model. Beliaeva also describes two other lattice-based procedures for option pricing (particularly American options) from literature. Like Tseng's method above, none of the procedures utilize recombining trees. In fact, Beliaeva claims that the recombination problem is the biggest issue with any stochastic volatility lattice or tree method, because non-combining trees require an exponentially rising number of nodes.

Leisen (2000) [12] developed a tree method for approximating continuous-time stochastic volatility models. This technique was general in the sense that it could be applied to different models, including those of Hull and White (1987) [11], Chesney and Scott (1989) [2], Stein and Stein (1991) [14], and Heston (1993) [8]. The lattice is constructed in a somewhat similar fashion as our method: construct a one-dimensional grid for the volatility process and extend it to a two-dimensional grid for the volatility/stock process. Therefore, each node of the two-dimensional grid has eight successors. However, there are two main weaknesses of the Leisen method compared to the Hull and White tree. First, there is no specific treatment of nodes that are at zero or below zero. Second, due to the lack of recombination, the procedure has high computational intensity, making it impractical if the number of time steps is too large.

Guan and Xiaoqiang (2001) [7] attempted to solve the recombination problem of stochastic volatility trees by using a weighted-average interpolation of nearby option prices to find the required options prices to calculate the tree. While node recombination is not an issue in this method, Guan and Xiaoqiang identified that there are particularly large errors with this method because of the interpolation, especially as term-to-maturity increases (since the interpolation error multiplies over the life of the option). The authors avoid this problem by focusing mainly on short term-to-maturity options.

Aside from the ones listed above, there exist other SV tree techniques in literature. Florescu and Viens [6] in 2005 used an algorithm taken from genomics to estimate the stochastic volatility distribution. They then construct a two-dimensional tree for each volatility level that is recombining but computationally unmanageable on its own. To solve this problem, Florescu and Viens select an n-sized bootstrap sample from the discrete volatility distribution and construct many much smaller trees for these volatility levels. They then compute the option value at the terminal nodes for each tree by using the usual binomial tree method, taking the average at the end to estimate the stochastic volatility option price. Of course, as with any Monte Carlo method, there exists a level of randomness (especially with a smaller bootstrap sample), a problem not encountered by our recombining tree.

Liu [13] in 2008 developed a similar method to ours but used a regime-switching volatility model instead of a stochastic volatility model. Regime-switching volatility models are stochastic in the sense that the volatility process in such a model is "stochastic" (i.e. it varies randomly in time). A regime-switching volatility model, however, utilizes a Markov chain to dictate asset prices, and model parameters such as drift and volatility coefficients depend on the Markov chain. Liu's technique also uses the binomial option pricing model to create a recombining tree that grows linearly as the number of time steps increases. The paper also demonstrates the method's strength in using a large N to accurately calculate option prices for both European and American options. However, this paper indicates that although the tree grows linearly with N, a higher number of regimes (volatility states) makes computation times impossibly high.

2 Description of Method

Before presenting the option pricing tree method that is our focus, it is first necessary to review stochastic volatility using Hull and White's trinomial tree approach. Then we will go over the initial assumptions of the model, followed by the methods that generate the volatility states, underlying prices, and option prices.

2.1 General Method

2.1.1 Approach and Assumptions

The main approach to this binomial-trinomial tree technique revolves around the possible prices that the underlying can take after positing a stochastic volatility model. We make two important assumptions for this method:

1. By choosing appropriate spacing between volatility states, we wish to minimize the number of possible prices in order to decrease computation time.
2. In order to use the binomial option pricing model without need for interpolation and other computationally intensive methods, we must make sure that the tree is recombining, even with stochastic volatility.

2.1.2 H&W Trinomial Tree for Volatility

Using Hull and White's Trinomial Tree method for stochastic volatility is similar to the method's original use for interest rates. In fact, the only parameter that changes is that m is no longer the initial interest rate; it is now the initial volatility of the returns of the underlying asset of the option. Using the Hull and White (1994) branching schemes and probabilities (Figure 1), we create a trinomial tree of volatilities, which is stochastic due to the "volatility of the volatility" β and is mean-reverting from the OU process.

However, when the method finds the option price at time zero, the volatility can be any value in the range between j_{max} and j_{min} (i.e. the starting volatility could theoretically be at any point in the range); therefore, a "tree" of volatilities (in the sense of starting at one node and branching out) is not actually necessary. At any time n, the volatility of the underlying asset can be any of the states between j_{max} and j_{min}. Therefore, all that is necessary for the method is a list of all possible volatilities, which is easily created without using a tree by the following: the volatility values are centered at m, are as high as $m + j_{max}x$ and are as low as $m + j_{min}x$. With Hull and White's method, we obtain this list of volatilities, which is crucial to actually pricing the option using the binomial option pricing model.

2.1.3 Recombining with Volatility

To find the volatility values that recombine the tree, we first start with a value c, which is the lowest possible volatility state, $m + j_{min}x$. Given that the tree must recombine, c must be a

multiple a of the spacing between volatilities x because of the up ($u = e^{\sigma\sqrt{\Delta t}}$) and down ($d = e^{-\sigma\sqrt{\Delta t}}$) coefficients in the binomial option pricing model. That is,

$$c = m + j_{min}x = m - j_{max}x = ax. \tag{5}$$

Solving for a in equation (5) gives

$$c = \frac{m}{x} - j_{max} = \frac{m}{\beta\sqrt{y\Delta t}} - j_{max}, \tag{6}$$

which indicates that m must be a multiple of x if the prices are to recombine. This means that Hull and White's recommendation of $y = 3$ will almost always be inadequate since we cannot guarantee that m is a multiple of x if $x = \beta\sqrt{3\Delta t}$ (if β and Δt are given parameters). Instead, we must find an acceptable and optimal value of y.

For y to be acceptable in accordance to equation (6), it must satisfy the following equation obtained from solving equation (6) for y:

$$y = \left[\frac{m}{\beta(a + j_{max})}\right]^2 (\Delta t)^{-1}. \tag{7}$$

But how do we select a value for a? First, note that from equation (5) that as a increases, x is smaller relative to c, which means that there would be more nodes in the final step of the S_t tree. Then to optimize computation time, we know that a must be as small as possible (while still at least 1) in order to minimize the number of nodes. If there were no constraints on y, then we could always achieve $a = 1$. But Coulon [3] tells us of two constraints on y in order for the H&W probabilities to hold:

1. $\frac{4}{3} < y < 4$, and
2. $1 - \sqrt{1 - \frac{1}{y}} \leq b \leq \sqrt{1 - \frac{1}{y}}$.

Equation (6) tells us that as y increases, a decreases. Therefore, the optimal value of y is its maximum value. To find this, we first set $y = 4$ in equation (6) and find the ceiling function of the resulting a value. From this a value, we then use equation (7) to find the value of y that minimizes computation time and recombines the tree (which should now be smaller than 4 since the ceiling function increases a). With this value of y and the other given parameters, Hull and White's trinomial tree method can generate an acceptable list of possible volatilities for this recombining tree.

2.1.4 Underlying Prices

The list of volatilities obtained in Section 2.1.2 gives insight on the possible prices for the underlying asset throughout the binomial tree. First we want to write an expression for the largest value in this list of prices. We start with the u of starting volatility σ_0 (which is equal to m in this case), $u_m = e^{m\sqrt{\Delta t}}$. Therefore, u_j of volatility $m + jx$ is

$$u_j = e^{(m+jx)\sqrt{\Delta t}} = u_m e^{jx\sqrt{\Delta t}}. \tag{8}$$

From this, the expression for the highest price is

$$S^{max} = S_0 \left(u_m e^{j_{max} x\sqrt{\Delta t}}\right)^N. \tag{9}$$

Likewise, replacing j_{max} with j_{min} in equation (9) gives us S^{min}, which is the lowest possible price. In both cases, max and min can be treated as positive and negative numbers, respectively, representing the index of each price in the complete price list.

The total number of prices is determined by the size of the volatility list, by the equation

$$2N\left(\frac{m}{x} + j_{max}\right) + 1, \tag{10}$$

where the 1 at the end is due to price S_0.

From equation (9), we see that the value of the second highest price is

$$S^{max} e^{-x\sqrt{\Delta t}}. \tag{11}$$

We can then perform equation (11) repeatedly to generate the number of prices specified in equation (10) to make the complete price list.

2.1.5 Option Price

After obtaining all of the possible prices of the underlying asset, calculating the option price is simply a multi-period binomial model for option pricing with the addition of an extra dimension, volatility (which is calculated at each step from the H&W probabilities from Figure 1). While working backwards in the multi-period binomial tree (pricing by discounted risk-neutral expectations), the option price at each node is just a modified form of the same equation for the CRR binomial option pricing model:

$$\begin{aligned} V_{j,s,n} = e^{-r\Delta t}\{ & q_u\left[qV_{j+z_1,s+\delta,n+1} + (1-q)V_{j+z_1,s-\delta,n+1}\right] \\ & + q_m\left[qV_{j+z_2,s+\delta,n+1} + (1-q)V_{j+z_2,s-\delta,n+1}\right] \\ & + q_d\left[qV_{j+z_3,s+\delta,n+1} + (1-q)V_{j+z_3,s-\delta,n+1}\right]\}. \end{aligned} \tag{12}$$

In this equation,

- j is the volatility index of the node.
- s is the underlying price index.
- n is the time period of the current node.
- q_u, q_m, and q_d are the probabilities obtained from Hull and White's trinomial tree, depending on the type of branching emanating from the current node (at j, n and s).

- q is the risk neutral probability obtained from the CRR binomial option pricing model $q = \frac{e^{r\Delta t} - d}{u - d}$, where u and d are obtained from the volatility state (using $u = e^{\sigma\sqrt{\Delta t}}$ and $d = \frac{1}{u}$) of the current node.
- z_1, z_2, and z_3 depend on the branching type. If the node has type (a) (normal) branching, $z_1 = \{1,0,-1\}$. If the node has type (b) (lowest volatilities) branching, $z_2 = \{2,1,0\}$. If the node has type (c) (highest volatilities) branching, $z_3 = \{0,-1,-2\}$.
- δ is a fixed increment $\frac{m}{x} + j_{max} - j + 1$.

As the tree steps back using equation (12), it fills in the three-dimensional tree with options prices as follows: S^{max} to S^{min} at $n = N$, $S^{max-\frac{m}{x}+j_{max}}$ to $S^{min+\frac{m}{x}+j_{max}}$ at $n = N-1$, $S^{max-2\left(\frac{m}{x}+j_{max}\right)}$ to $S^{min+2\left(\frac{m}{x}+j_{max}\right)}$ at $n = N-2$, etc. Note that this method calculates some "unreachable" nodes in the middle of the range; this will be analyzed in section 6.1. Once the tree reaches $n = 0$ (and $T = 0$), the tree will have a list of initial option prices V_0, one for each possible volatility state.

3 Analysis and Performance

This section will discuss how different parameters--- y and N in particular---affect the performance of the tree and the resulting option price. We will also study the effect of correlated volatilities on the option price calculated by the tree technique. For consistency, we will use the following Google call option (data from Yahoo! Finance [16]) to perform analysis for the rest of this paper. The parameters are listed below:

- Mean-reversion rate: $\alpha = 4$
- Volatility of volatility of returns: $\beta = 0.3$
- Mean volatility of returns: $m = 0.35161$
- Time (years) to expiry: $T = \frac{167}{365}$
- Number of periods in tree: $N = 30$
- Risk-free rate: $r = 0.0004$
- Initial price of stock: $S_0 = 642.92$
- Strike price: $K = 650$
- Parameter suggested by H&W: $b = 0.184$
- Price of option calculated with above parameters: $V_0 = 58.9845$
- Market price on 12/30/11 for reference: \$51.10/\$51.80

There is no market data available for the mean-reversion rate α and the volatility of the volatility β, so we used the following algorithm to estimate the parameters for the method:

1. Analyze the historical volatility of the underlying asset and decide what would be a reasonable 95% confidence interval for volatility (since $\frac{\beta^2}{2\alpha}$ represents the long term

variance of σ_t of the underlying asset [10]), and find a pair of α and β values that satisfy this interval.
2. Generate the volatility smile as in Figure 5 and check if it is acceptable; if not, change α and β until a smooth volatility smile is seen.
3. Check the list of possible volatilities generated by the Hull and White trinomial tree for zero and/or negative values. Adjust the parameters accordingly.
4. (Optional) After the first three conditions are met, increase α and β within a reasonable range (assuming no market data) to decrease computation time.

3.1 Changing Parameters

As y increases, the coefficient a in $c = ax$ decreases; this in turn decreases the size of the problem by reducing the number of possible prices that the underlying asset can reach. Further, according to Coulon [3], one of the conditions that y has to follow is $\frac{4}{3} < y < 4$. Since prices are generally shown vertically in a binomial pricing tree, we will call this the "vertical size" of the problem. If we drop assumption 1 (to minimize the size of the problem) and decrease y to observe its effects, y must still stay in the above range. As an additional constraint, since the values of y must be such that its corresponding a is an integer, there are only a few possible values of y that satisfy both this condition and $\frac{4}{3} < y < 4$. To observe the effects of changing y on the computation time and the option price, we again use the call option on Google and use the price obtained for the initial volatility state $m = 0.35161$. The computational time is averaged (across five runs) due to the fluctuations in the time it took for MATLAB to run the method. From the table in Figure 2, we note that although y does have a significant effect on computation time (about 33% in the example), the computation time is the minimum when y is the largest, which was our assumption to begin with. This is due to the fact that each increase in a increases the number of possible stock prices by a constant (40 in this case). Also, note that changing y does not significantly change V_0.

For N, the number of periods in the pricing model, it is intuitive that increasing N would increase the computation time because of the outward branching of binomial trees. Because the size of N is represented horizontally in such a tree, we will call this the "horizontal size" of the problem. Again, using the same Google call option, we observe the effect of changing N in the middle plot in Figure 2. The graph shows that N has a substantial effect on the technique's computation time. From this analysis, it seems as if the tree runs in $O(N^3)$ time; for this particular option, $t = 0.0002N^3 - 0.0083N^2 + 0.1514N - 0.6775$ fits with an R^2 of 0.9928. This is true because increasing N increases the size of the tree in all three of its dimensions. Firstly, the number of time steps obviously increases. Secondly, the number of volatility states increases because since $j_{max} = \left\lceil \frac{b}{\alpha \Delta t} \right\rceil$, j_{max} is inversely related to Δt, which means it increases linearly with N. Lastly, increasing N increases the number of possible prices linearly by the equation $2N\left(\frac{m}{x} + j_{max}\right) + 1$.

To better observe the effect of changing N on V_0, we decreased the increments between each observation of N to produce the bottom plot in Figure 2. From this figure, it is clear that as N increases, V_0 converges---though it fluctuates due to the alternating effect of tree methods (see

Section 6.2). The dark line represents the Black-Scholes price of the option, and the converging price is higher than this value.

Now that we have analyzed the different effects of y and N on the vertical and horizontal sizes of the problem, we can look to revise our assumptions from when we first described the method. Although increasing y to a value as close to 4 as possible does decrease the vertical size of the problem, this effect is negligible in computation time when compared to decreasing N to decrease the horizontal size. This suggests that we may be able to decrease y to improve other aspects of the method, such as the price of V_0. However, we also found that changing y does not actually cause significant variation in V_0, so y may not play an important role in this problem.

3.2 Correlation

Many people working in finance would agree that volatility of an asset is not fully independent of the asset; instead, it often increases as the price decreases. We observe this when W and W^* from the SDEs in this method are not independent. As such, to account for this correlation, we adjust the Hull and White probabilities from Figure 1 with a value ε in order to factor in the volatility's negative correlation with stock price:

	σ_u	σ_m	σ_d
u	$q_u q - \varepsilon$	$q_m q$	$q_d q + \varepsilon$
d	$q_u(1-q) + \varepsilon$	$q_m(1-q)$	$q_d(1-q) - \varepsilon$

(13)

Here, q, q_u, q_m, q_d are the same as those from the complete equation for the option price (equation (12)).

Choosing this ε is somewhat arbitrary, but the main constraint is that after factoring it in, the probabilities still have to stay in between zero and one. This is the lattice feasibility problem discussed in Tseng et al.'s [15] two-factor method. So in order to pick ε, we must calculate the values from the table above and find the limiting factor (the probability that will most easily break the range of zero to one). To expedite this process, Hull and White's [9] studies state that the probabilities of the non-normal branching types (at j_{max} and j_{min} are the limiting factors in this case. Therefore by calculating the extreme values for q, q_u, q_m, and q_d, we can find the highest possible ε for our Google call option across different N values. With these ε values, we perform analysis in the top plot of Figure 3 and note that adding correlation into our tree method decreases V_0 across a wide range of ε values. This is intuitive because empirically, negative correlation between an asset's volatility and its price decreases the price of a call option. Note that as correlation increases, its effect on the price of the option also increases.

In this section, we presented a relatively narrow viewpoint of the effect of correlation on the option. We will perform a more comprehensive analysis of correlation in Section 5.3, since the effect of correlation is better visualized through the implied volatility curve.

3.3 Comparison to Monte Carlo Simulation

Another way to assess the quality of the solution of this method is to compare it to one of the most commonly used methods of option valuation for stochastic volatility, Monte Carlo

simulation. Again, this section uses the Google stock example. To perform the simulation, we discretize the original two SDEs as follows:

$$\Delta S_t = rS_t\Delta t + \sigma_t S_t \Delta W_t, \tag{14}$$

where $\Delta W_t = \sqrt{\Delta t}Z$. Z is taken from a standard normal $Z \sim N(0,1)$.

$$\Delta \sigma_t = \alpha(m - \sigma_t)\Delta t + \beta \Delta W_t^*, \tag{15}$$

where $\Delta W_t^* = \sqrt{\Delta t}Z^*$. Z^* is taken from a standard normal $Z^* \sim N(0,1)$ that is independent of Z.

From equation (14), we start with the given S_0 and $\sigma_0 = m$ to calculate S_1. Then we use equation (15) to find σ_1, which is then paired with S_1 to find S_2. We repeat this process until $t = T$ to simulate one path of the underlying asset, with which we can calculate V_0 by discounting the payoff of S_T. After creating N^{sim} paths, we find the option price by averaging the discounted payoffs.

By using the Google call option from before, we analyzed the effect of the number of simulations N^{sim} on the run time and the price convergence of the option. We set $N = 50$ since that was largest N used in the tree method. As the number of paths N^{sim} increases, the simulation time increases linearly. To better compare the price convergence of Monte Carlo simulations and our recombining tree method, we plotted the prices of both methods against computation time in the second plot Figure 3 (the black line again represents the Black-Scholes price of the option), which shows an important result. The tree method strongly outperforms Monte Carlo simulations of the option price when compared by computation time, in both convergence speed and accuracy (since simulation produces different paths each time). Both methods return an acceptable average (for simulation, this average is weighted by N^{sim}) value over all of the points; simulation results in \$58.75 and the tree results in \$58.98. However, the standard deviation for the tree method is \$0.09, which is significantly smaller than the simulation's \$1.88. Note also that $N = 50$ for all of the simulation results, while the tree here only uses N values ranging from 5 to 50 (in order to match the computation times as closely as possible); this makes the result even more significant. Even at $N = 50$ (the rightmost white point), the stochastic volatility tree uses less than half of the computation time of the largest simulation we sampled ($N^{sim} = 15{,}000$).

To further verify the pricing efficacy of the tree method, we ran a large simulation with $N^{sim} = 1{,}000{,}000$ and used this price as a benchmark for the "true" value of the Google option given our inputs. The simulation returned a price of \$58.76 with a 95% confidence interval of (\$58.56, \$58.97) whereas the tree method converged to roughly[3] \$58.90 well before N even reaches 50. However, since Monte Carlo simulation's error that decays at a rate of $\frac{1}{\sqrt{N^{sim}}}$ (quite slow), this is another instance of our tree method outperforming simulation. Combined, these results show that the tree method is not only superior to the Monte Carlo simulation computationally, but also is effective in correctly valuing options with a faster price convergence.

[3] Because of the alternating effect of tree methods (see Section 6.2), the tree method never truly converges but instead jumps up and down (in very small amounts) around this price.

4 American Options

American options are often difficult to price explicitly or via simulation because of the early exercise decision. However, the recombining tree approach makes pricing American options very straightforward: we merely have to check for an early exercise at each node calculated in the option value tree. Therefore, the run time difference between American and European options is negligible, and this is indeed one of the main reasons to use a tree method for American options.

4.1 Validity

To test the validity of the technique when pricing American options, we refer back to the same Google call option as before, except this time we will model it as an American option. Since Google does not pay dividends, we can expect the prices of the call to be the same for both American and European calls. For puts, we use the same parameters (the put also has a strike of $650) as in the call and we now expect the American option to cost more than the European in every starting volatility state because of the early exercise option. The stochastic volatility tree confirms both hypotheses.

4.2 Optimal Exercise Boundary

Another important aspect of American options is the optimal exercise boundary. To study this, we will use the same Google stock, but instead with a put option with strike $K = \$600$ (other parameters of the option stay the same). The first plot in Figure 4 shows the boundary with the highest, lowest, and mean (m) volatility states as σ_0. The second plot shows the results of changing the volatility of the volatility, β. To avoid confusion, we only plot the boundary for volatility $\sigma_t = m = 0.35161$ for this case. These results make intuitive sense because at low volatilities, one is more likely to exercise the put at a higher price since there is a smaller probability that the stock price will fall deeper into the money. In addition the value of β has a small but noticeable effect on the early exercise boundary: an increase in the volatility of the volatility does shift the boundary down slightly. In addition, it seems as if a higher β leads to a later optimal exercise boundary. This result is consistent with the theory behind stochastic volatility models: stochastic volatility causes fatter tails in the distribution of $\log S_t$ and increasing β can be thought of as making the process "more stochastic" since the volatility has a larger range of values with increasing β. With these fatter tails, one is less likely to exercise an American put early.

4.3 Correlation

As discussed in Section 3.2, correlation often plays a large role in stochastic volatility. To see how this affects the American options, we plotted the optimal exercise boundaries for the same put option from above with three different correlation levels (different ε values). Again, we only

plot the boundary for volatility $\sigma_t = m = 0.35161$ for each case. The third plot in Figure 4 shows that an increase in correlation shifts the optimal exercise boundary upward for this put option; this is consistent with the effect of correlation in Section 3.2 decreasing the option price.

5 Implied Volatility Surface

Since options trading is so prevalent in the finance industry today, the implied volatility surface of derivatives is becoming more and more important as a means for traders to predict the price of an option as its parameters change. In this section, we will first show that our stochastic volatility tree produces a volatility smile (which often appears empirically in markets) and then we will analyze the effect of various parameters on the shape of the curve.

5.1 Volatility Smile

In the previous sections, we used an example of a real-world Google option to perform analysis. Here, we will show the implied volatility curve (vs. strike) for the Google call. In the first plot of Figure 5, the black line shows m, the initial (mean) volatility. The plot of implied volatility against strike shows consistency with the market in three important ways that demonstrate the strength of the stochastic volatility tree:

1. There is a clear volatility smile.
2. Implied volatility is the lowest at-the-money.
3. Implied volatility is always higher than m due to the stochastic nature of the volatility.

5.2 Changing parameters

As discussed earlier, one of the factors we consider when choosing appropriate α and β parameters is whether or not the tree generates a smile with the chosen parameters. The second plot of Figure 5 shows this effect with the Google stock call option. Holding α constant at 4, we decrease β from 0.3 and observe the effects. As β decreases, the volatility curve becomes less erratic and approaches Black-Scholes; in fact at very low β, the mean-reversion of the tree even pulls the volatility smile into a volatility "frown." Another observation we can make is that as the effect of β decreases, the alternating effect of tree methods (Section 6.2) is more apparent.

Another value that affects the shape of the volatility curve is the number of time steps N. The first plot of Figure 6 shows the effect of three different N values on implied volatility. It is apparent that the volatility smile becomes steeper as N increases. This occurs because increasing N decreases Δt, which increases j_{max} which causes more extreme volatility states; this gives the $\log S_t$ distribution even fatter tails. Another effect of increasing N is the diminishing of the alternating effect of tree methods (this smoothens the volatility smile). This is especially apparent near the at-the-money strike, as we can see from the $N = 15$ plot. These results show that N is not a parameter like the ones above, since very low values of N yields a poor model with our tree method.

5.3 Correlation

Section 3.2 showed us that correlation had a substantial effect on the price of the option, so we can expect the same effect to carry over to the implied volatility curve. A comparison of the original smile and one with the maximum correlation (as calculated in Section 3.2) is shown in the second plot of Figure 6.

In Section 3.2, we only looked at the effect of correlation on a call with a strike $K = 650$. This plot presents a better representation of why the price of the option decreased at that particular strike. However, the graph also shows that for calls with lower strikes (below $K = 620$), correlation actually increases the price of the option and makes the smile steeper. This is consistent with the empirical data on the negative correlation between strike and volatility: volatility tends to increase as the strike decreases, and vice versa.

6 Further Study

Finally, in this section, we will look at how the method could possibly be improved, followed by other applications of the technique.

6.1 Computation Optimization by "Gaps"

In any method for options pricing (and in most other fields), computation speed is of utmost importance due to the fast-paced nature of markets. With this particular method, there is one aspect that can potentially be improved because it causes unnecessary calculations when pricing an option. To thoroughly explain this, we present the following example:

1. A stock starts at price S_0 and there are three possible volatilities at each time step. The complete price array consists of all values, inclusive, between $S^{max}$and S^{min}.
2. The volatilities are such that at step $n = 1$, the possible prices for S_1 are S^j, S^{j+1}, S^{j+2} for up states and S^j, S^{j-1}, S^{j-2} for down states. It is apparent that if $j > 0$, then there are "gaps" between the possible prices at $n = 1$ where S_1 cannot possibly reach.
3. These gaps will decrease at every time step and eventually close if N is large enough.

The recombining tree technique does not take these gaps into account. Instead, it merely truncates the price array at the top and bottom with each step backwards in time. Therefore, at the earlier time steps, the method may calculate option prices for nodes in the middle of the tree that are not actually reachable by the underlying price tree. If this calculation time is significant, it could impact the total run time of the tree technique. To better observe how much these gaps matter, the first two plots of Figure 7 show the total number of possible prices at each step of pricing the Google call option plotted along with how many prices the method actually calculates (the first one is for normal parameters, the second is for a larger a).

In the first plot of Figure 7, periods $10 < N \leq 20$ are omitted in order to better visually observe the difference between the two lines. As expected, the gap in this case quickly closes and the method only performs 19 (out of 7161 necessary calculations, or about 0.3%) more calculations than necessary compared to the actual possible prices. However, if the coefficient a

in $c = ax$ increases--that is, if β decreases, we expect the gap to take longer to close. This result is represented in the second plot of Figure 7 (note the scale is $1 < N \leq 20$) for $\beta = 0.025$ (instead of 0.1). In this case, we perform 1335 extra calculations (out of 21861, or about 6.1%), which is significant enough to increase the run time.

We could not find a method to perform only the calculations required without sacrificing the speed of the tree method described. In fact, the method to find the actual number of possible prices was extremely computationally intensive and cannot be practically implemented for large N. At each period n, it branches out the prices from the previous period, removes the duplicates, and repeats until it finds the complete price array at period N. Therefore, theoretically, the method's computation time can be optimized more, but realistically, it may not be possible.

6.2 Alternating Effect of Tree Methods

One of the downsides of using any tree method for valuing derivatives is that the price of the underlying asset is always, to some extent, discretized. Because of this, the strikes of such options are never exact, and how close to the strike is to its closest price node at any given time period $N = n$ has a significant effect on the price of the option. This alternating effect can be seen in the third plot of Figure 7, which shows how the price hits peaks and troughs depending on whether N is odd or even.

Even though this alternating effect exists in something as simple as a vanilla call option, it is especially prevalent in barrier options, when an option can become worthless after hitting an exact barrier B. When B is close to S_0, the significance of the alternating effect of tree methods increases, since even a small "cushion" due to the closest price node not being exactly the value of B can change the probability of hitting the barrier by a considerable amount. One can think of the magnitude of this undesirable effect as being proportional to the magnitude of the Gamma of an option.

To overcome this weakness, we must increase the concentration of price nodes within a set price range to make the tree as continuous as possible. Increasing N alleviates this problem, as we can see in the third plot of Figure 7; however, this has the high cost of increasing computation time by an order of N^3. Two other ways to reduce the alternating effect of tree methods are decreasing α and increasing β. While both mitigate the alternating effect of tree methods, increasing β has a much larger effect. In other words, this stochastic volatility tree method performs better for assets with higher β. Decreasing α from 10 to 4 while keeping β constant barely decreases the amplitude of the alternating waves, whereas increasing β from 0.3 to 0.4 has a significant effect on the size of the alternating effect. However, like N, changing α and β also has its costs. The parameters are important in determining how similar the underlying asset is to a real-world asset, so changing them may affect whether or not such a mean-reversion or volatility of volatility is realistic.

6.3 Other Applications and Conclusion

This recombining stochastic volatility tree method for options pricing presents results that are competitive with more conventional methods such as Monte Carlo simulation in both speed and accuracy. It has even more potential if the computation time is optimized in ways such as the one presented in Section 6.1. In addition, further analysis showed that the tree method accurately

estimates the Greeks and options prices for first generation exotic options. We also performed successful analysis with commodities options by implementing a mean-reverting price (in addition to a mean-reverting volatility) stochastic volatility tree, since unlike traditional financial assets, commodity spot prices are often assumed to mean-revert. This extension came with the added benefit of improved computation speed because the number of nodes grew more slowly with increased N.

Since Cox, Ross, and Rubinstein showed that a tree model was a reasonable way to value financial derivatives, tree methods have been studied in great detail and more recently in conjunction with stochastic volatility. Given the prevalence of option trading in financial markets today, academics are constantly trying to find better and faster ways to value the derivatives. Even though this method is limited by the same weaknesses that affect all tree methods for option pricing, it shows its viability in the financial markets in its relatively fast computational time as well as results that are consistent with well-known properties of stochastic volatility models. With further research, this technique could be improved to be usable for a wider range of parameters, a greater set of derivatives, and more general stochastic processes.

References

[1] Beliaeva, Natalia A. “Efficient Lattice Methods for Pricing Contingent Claims Under Stochastic Volatility and Jumps Models.” Ph.D. thesis, University of Massachusetts Amherst, 2006.

[2] Chesney, Marc and Scott L. “Pricing European Currency Options: A Comparison of the Modified Black-Scholes Model and Randon Variance Model.” Journal of Financial and Quantitative Analysis 24: 267-284. 1989.

[3] Coulon, Michael. “Modeling Price Dynamics Through Fundamental Relationships in Electricity and Other Energy Markets.” Ph.D. thesis, Oxford University, 2009.

[4] Cox, John C., Stephen A. Ross, and Mark Rubinstein. “Option Pricing: A Simplified Approach.” Journal of Financial Economics 7: 229-263. 1979.

[5] Dumas, Bernard, Jeff Fleming, and Robert E. Whaley. "Implied Volatility Functions: Empirical Tests." The Journal of Finance 53.6. 1998.

[6] Florescu, Ionuţ and Frederi Viens. “A binomial tree approach to stochastic volatility driven model of the stock price.” Annals of University of Craiova, Math. Comp. Sci. Ser. 32: 126-142. 2005.

[7] Guan, Lim Kian and Xiaoqiang Guo. “Pricing American Options with Stochastic Volatility: Evidence from S&P 500 Futures Options.” Journal of Futures Markets 20, N7: 625-659. 2001.

[8] Heston, Steven L. “A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency.” The Review of Financial Studies 6: 327-343. 1993.

[9] Hull, John C. Options, Futures, and Other Derivatives. Upper Saddle River, NJ: Prentice Hall, 2009. Print.

[10] Hull, John C. and Alan White. “Numerical Procedures for Implementing Term Structure Models I: Single-Factor Models.” Journal of Derivatives, 2(1), 7-16. 1994.

[11] Hull, John C. and Alan White. “The Pricing of Options on Assets with Stochastic Volatility.” Journal of Finance 42: 281-300. 1987.

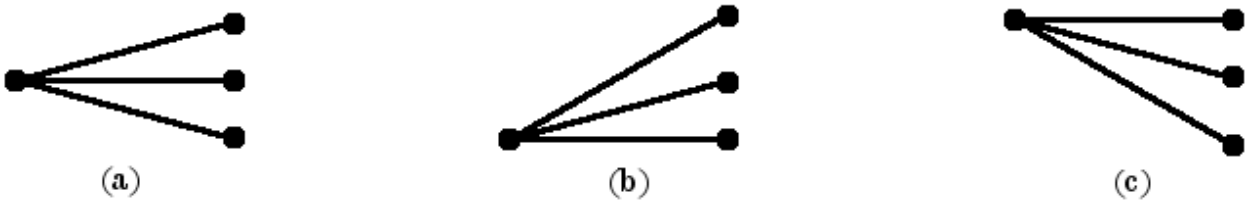
[12] Leisen, Dietmar P. “Stock Evolution Under Stochastic Volatility: A Discrete Approach.” Journal of Derivatives Winter: 9-27. 2000.

[13] Liu, Ruihua. “Regime-Switching Recombining Tree for Option Pricing.” International Journal of Theoretical and Applied Finance 13: 479-499. 2009.

[14] Stein, E.M. and J.C. Stein. “Stock Price Distributions with Stochastic Volatility: An Analytic Approach.” The Review of Financial Studies 4: 727-752. 1991.

[15] Tseng, Chung-Li, San-Lin Chung, and Pai-Ta Shih. “A General Two-Factor Lattice Method for Stochastic Volatility Models with Analysis of Lattice Feasibility.” Unpublished manuscript. Australian School of Business, The University of New South Wales. 2011

[16] Yahoo Finance! 2012. Yahoo Inc. 26 March 2012 <http://finance.yahoo.com>



Type (a) branches:

$$q_u = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 - \alpha j \Delta t) + \frac{1}{2y}$$

$$q_m = -\alpha^2 j^2 (\Delta t)^2 + 1 - \frac{1}{y}$$

$$q_d = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 + \alpha j \Delta t) + \frac{1}{2y}$$

Type (b) branches:

$$q_u = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 + \alpha j \Delta t) + \frac{1}{2y}$$

$$q_m = -\alpha^2 j^2 (\Delta t)^2 - 2\alpha j \Delta t - \frac{1}{y}$$

$$q_d = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 + 3\alpha j \Delta t) + 1 + \frac{1}{2y}$$

Type (c) branches:

$$q_u = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 - 3\alpha j \Delta t) + 1 + \frac{1}{2y}$$

$$q_m = -\alpha^2 j^2 (\Delta t)^2 + 2\alpha j \Delta t - \frac{1}{y}$$

$$q_d = \frac{1}{2}(\alpha^2 j^2 (\Delta t)^2 - \alpha j \Delta t) + \frac{1}{2y}$$

Figure 1: Different H&W Trinomial Branches and Probabilities

y	a	Time (sec)	Price at $T = 0$ (V_0)	Number of Possible Stock Prices
3.603	1	2.240	\$58.985	541
2.502	2	2.425	\$58.986	601
1.838	3	2.616	\$58.982	661
1.407	4	2.987	\$58.976	721

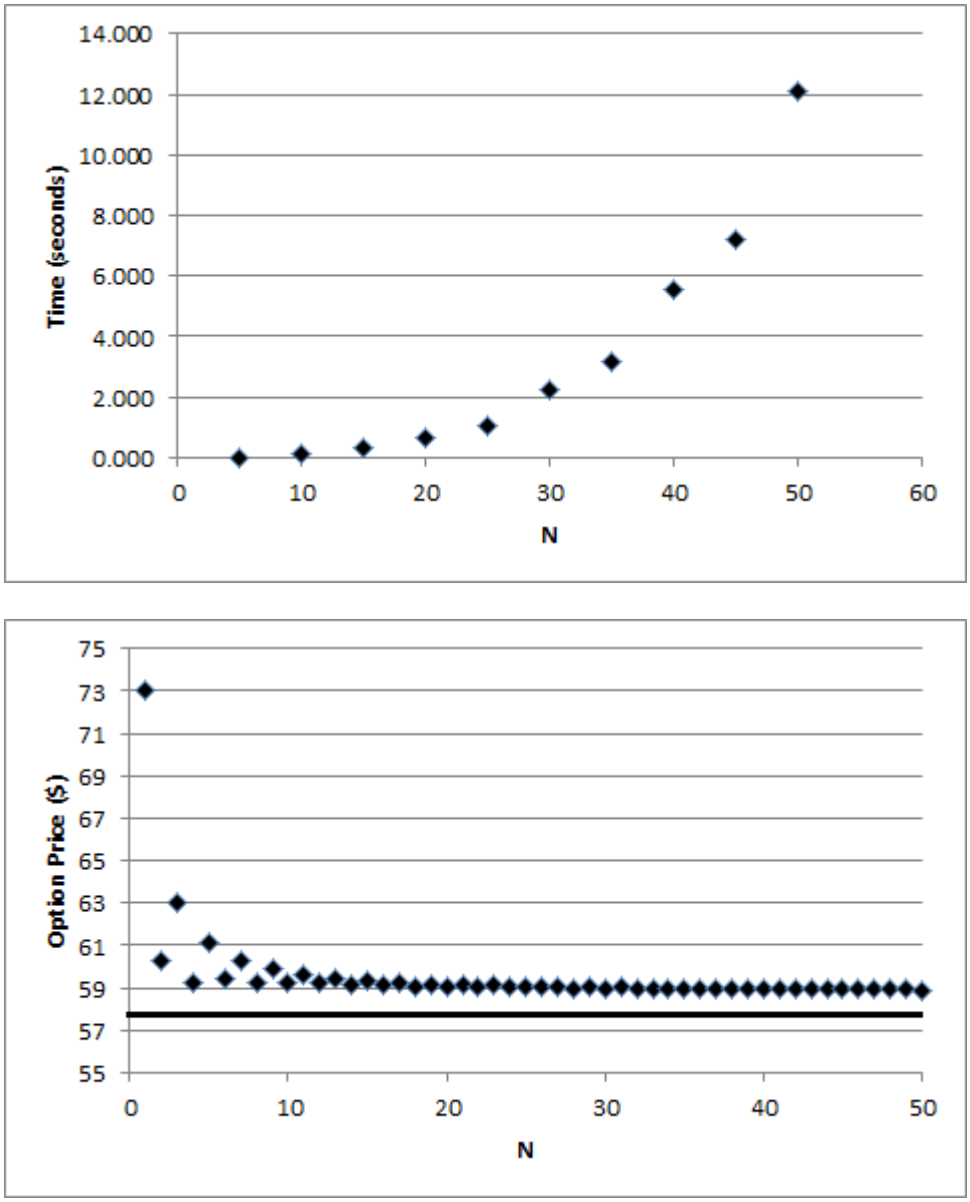
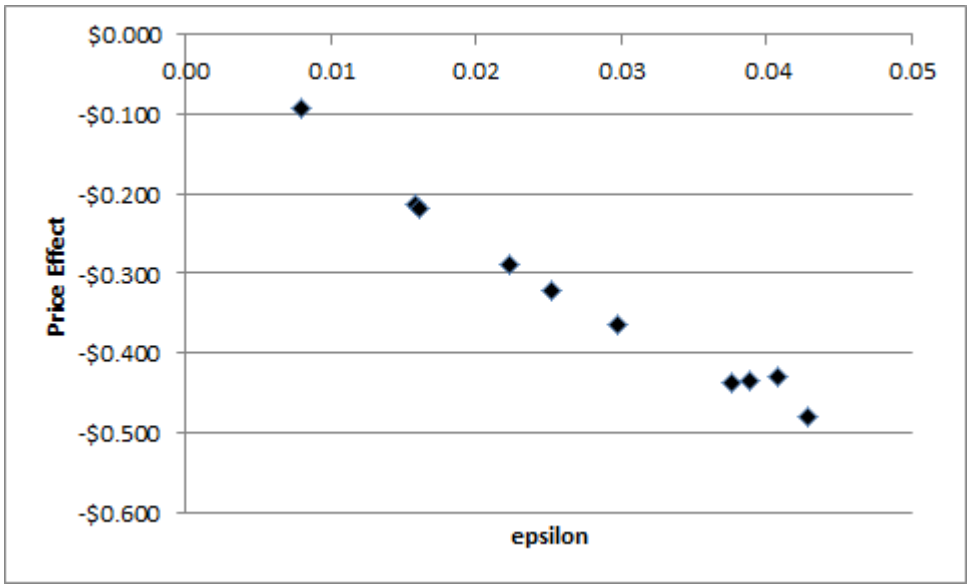


Figure 2: Effect of Changing Parameters on Tree Method



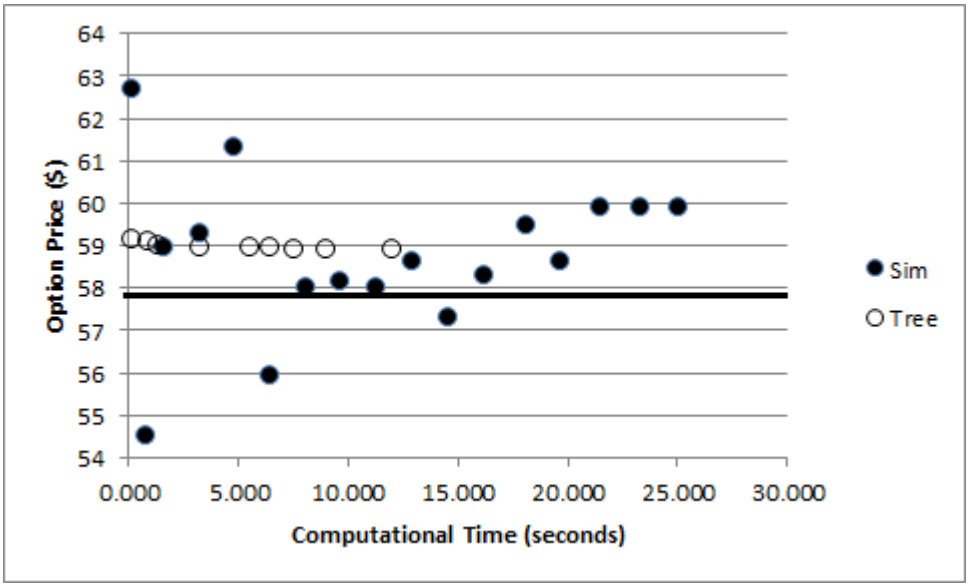
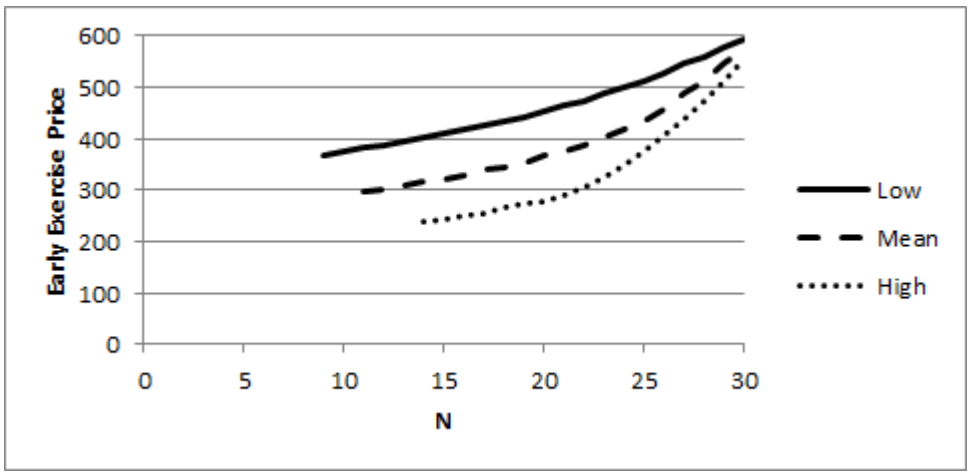
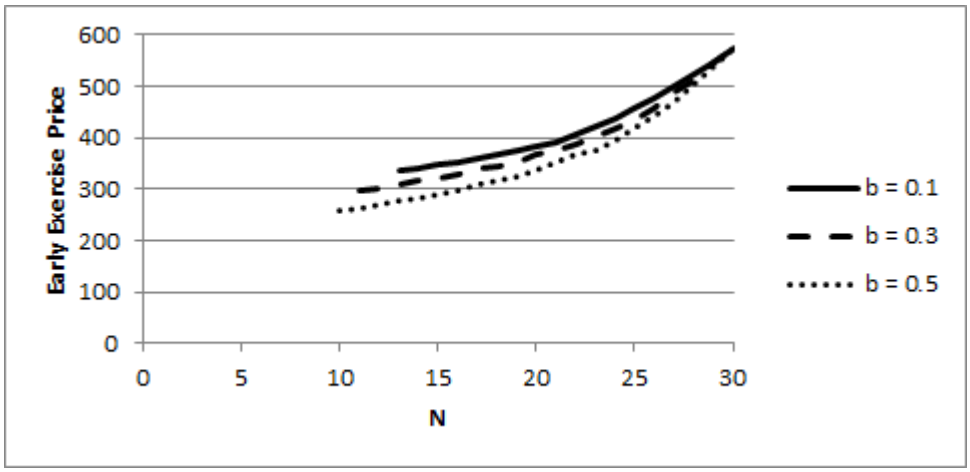


Figure 3: Correlation and Comparison to Monte Carlo Simulation





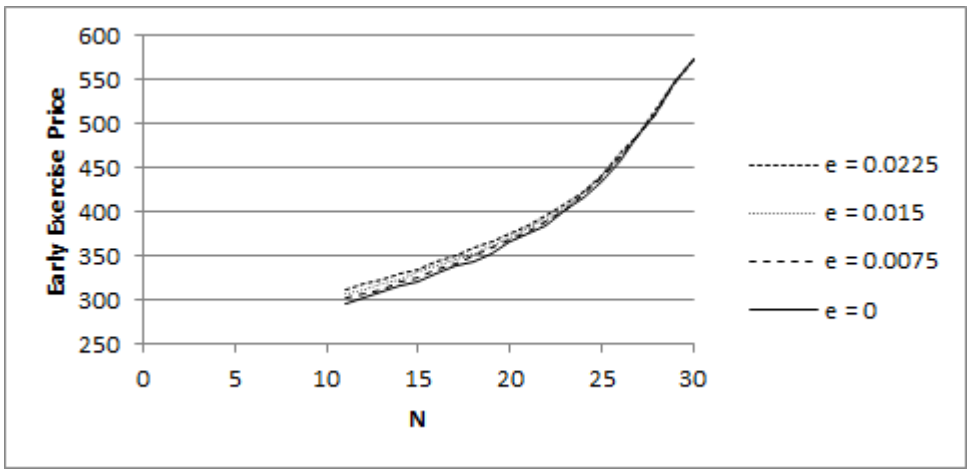
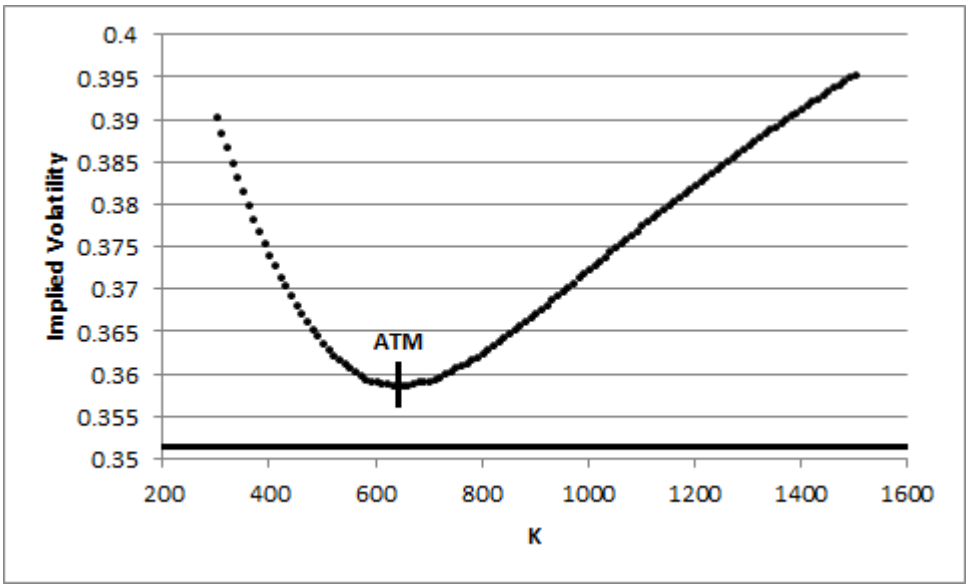


Figure 4: Optimal Exercise Boundary of American Put for Varying σ, β, and ε



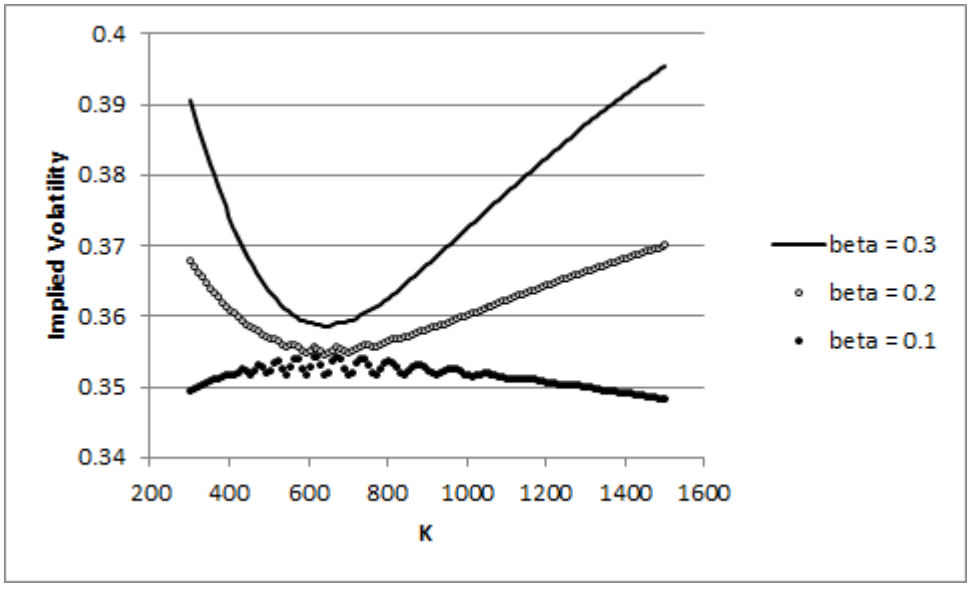
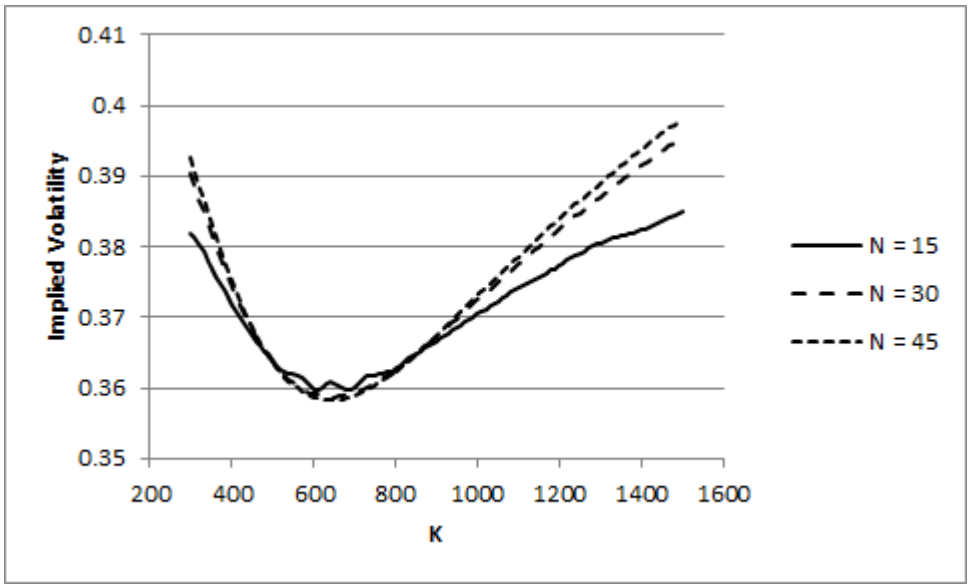


Figure 5: Implied Volatility for Google Stock Option with Varying α and β



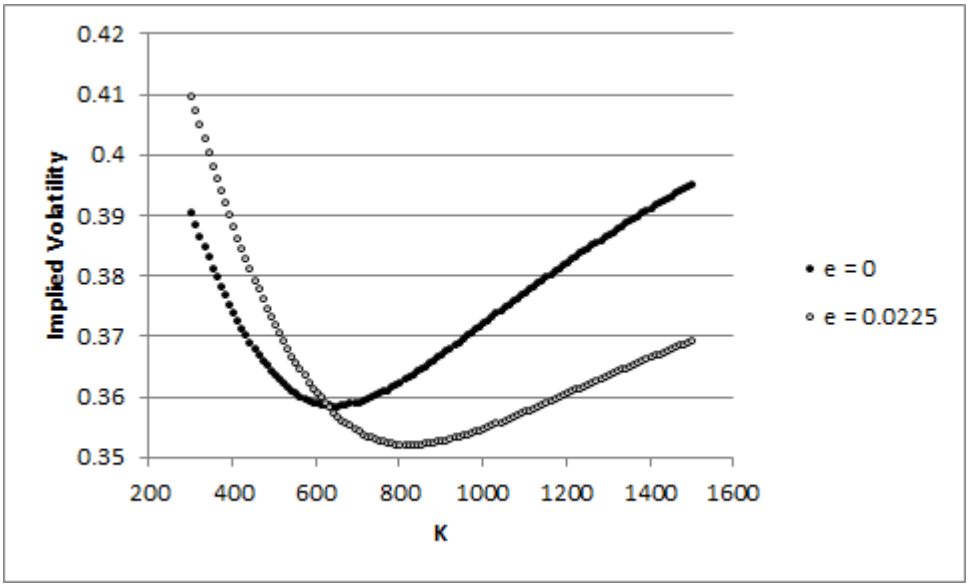
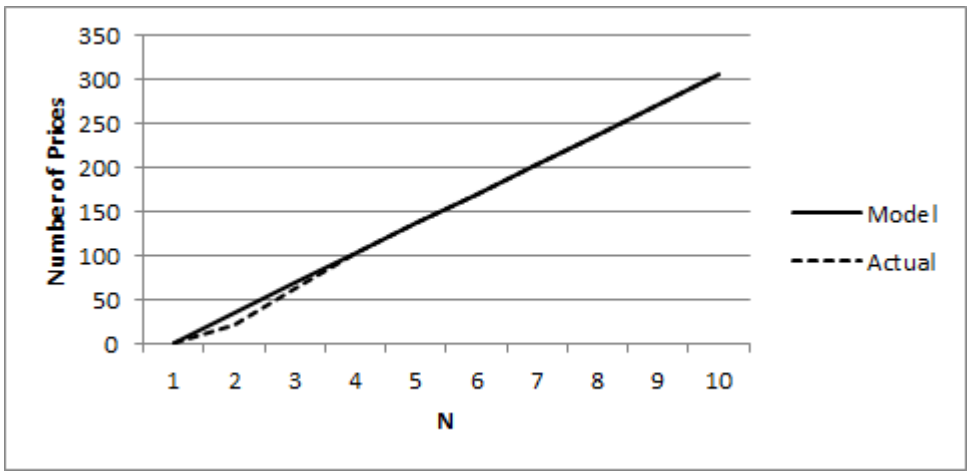
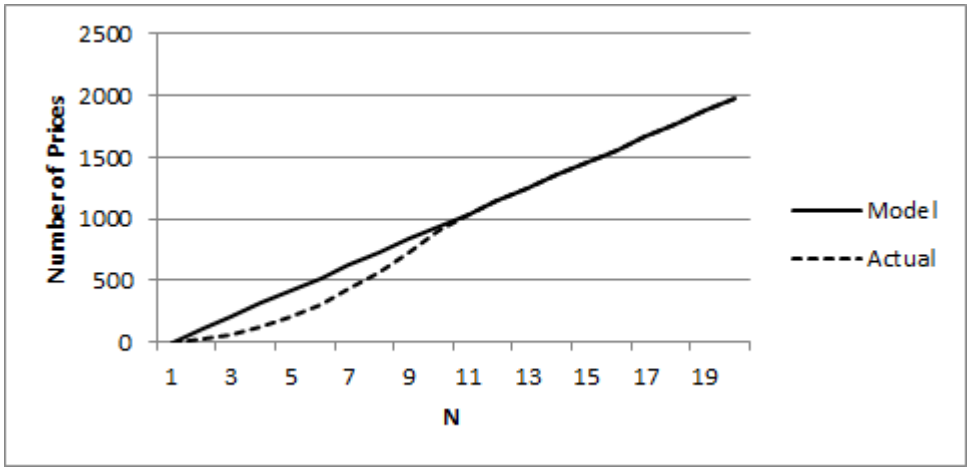


Figure 6: Implied Volatility vs. N and Correlation





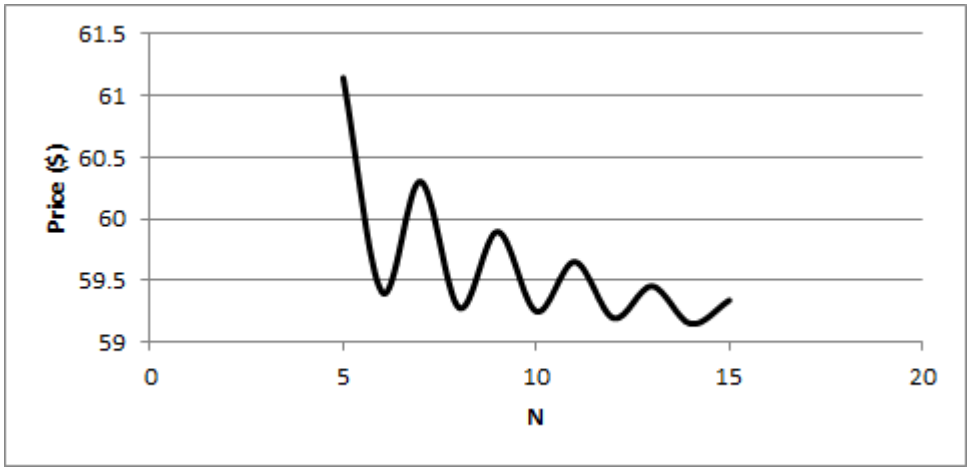


Figure 7: Visual Representation of "Gaps" and the Alternating Effect