DISTRIBUTION-BASED OPTION PRICING ON LATTICE ASSET DYNAMICS MODELS

YUJI YAMADA*
Graduate School of Business Sciences, University of Tsukuba
3-29-1 Otsuka, Bunkyo-ku, Tokyo 112-0012, Japan

JAMES A. PRIMBS†
Management Science and Engineering, Stanford University
Terman Engineering Center 700-327, CA 94305-4026, USA

Received
Accepted

ABSTRACT

In this paper, we propose a numerical option pricing method based on an arbitrarily given stock distribution. We first formulate a European call option pricing problem as an optimal hedging problem by using a lattice based incomplete market model. A dynamic programming technique is then applied to solve the mean square optimal hedging problem for the discrete time multi-period case by assigning suitable probabilities on the lattice, where the underlying stock price distribution is derived directly from empirical stock price data which may possess “heavy tails.” We show that these probabilities are obtained from a network flow optimization which can be solved efficiently by quadratic programming. A computational complexity analysis demonstrates that the number of iterations for dynamic programming and the number of parameters in the network flow optimization are both of square order with respect to the number of periods. Numerical experiments illustrate that our methodology generates the implied volatility smile.

Keywords: Incomplete markets, Lattice models, Minimum mean square hedges, Dynamic programming, Network flow optimization

1 Introduction

Option pricing theory has been the core of modern mathematical finance since the derivation of the famous Black-Scholes formula [4] which provides a theoretical value for European call/put options. At the heart of the Black-Scholes derivation are the following two key assumptions. The first is that the underlying stock price follows a geometric Brownian motion, implying that its log return distributions are

*E-mail: yuji@gssm.otsuka.tsukuba.ac.jp
†E-mail: japrimbs@stanford.edu
Gaussian. The second is the assumption that trading may take place in continuous time. However, these assumptions fail to hold in practice, because empirical stock prices are not Gaussian and their marginal distribution usually possesses heavy tails (e.g., [19, 24]). Moreover, trades can not be placed continuously in real markets. This discrepancy may results in differences between the Black-Scholes price and the true option’s price, and appears as the “smile effect” or “implied volatility smile” in real markets (see Chapter 17 of [15] and references therein).

Based on empirical market behavior, option valuation techniques have been extended to more realistic assumptions for the underlying stock process (e.g., [6, 15, 16, 20]) and markets (e.g., [13, 25, 26]). In particular, research on pricing models which takes into account the smile effect and/or heavy tail phenomena has become active in recent years [8, 9, 17, 21, 22, 23] with empirical evidence that implied volatility increases for in-the-money or out-of-the-money options. In this body of literature, Hull and White [15] developed a stochastic volatility model and showed that the price of a call option is given by the Black-Scholes price integrated over the distribution of the mean volatility under the assumption that the underlying asset price and its volatility are uncorrelated. One can show that the stochastic volatility model produces heavier tails than those for Gaussian distributions and that it generates an implied volatility smile under certain conditions. Also note that Avellaneda, Levy and Paras [1] presented an uncertain volatility model with a connection to the volatility smile, where the volatility is uncertain but is known to lie in a certain bound. On the other hand, Jarrow and Rudd [16] showed that the price of an option depends on the kurtosis parameter if the distribution is heavy tailed, by approximating the risk neutral probability density function by the “true” distribution. A related formula is also presented by Potters and Bouchaud [22], where a cumulant expansion is applied to explain the relation between the volatility smile and the kurtosis parameter. Although there is an extensive body of research on the subject of heavy tailed stock markets and smiley options, we will not be able to cover all of them, and we hope that readers will excuse our blatant omission of much of that work. For related work, see e.g., [10] and [15] and references therein.

The objective of this paper is to provide a numerical option pricing method for an arbitrarily given stock distribution, where we formulate the option pricing problem as an optimal hedging problem (e.g., [13, 25, 26]) and solve it on a lattice-based incomplete market model. An important feature of the optimal hedging problem formulated in this paper for an incomplete market model is that the price of a call option may depend on its real probability distribution (see Section 2 for the precise problem statement). In this paper, we propose an option pricing methodology which takes advantage of this dependence on real probabilities, especially for heavy tailed distributions. Here we solve the minimum mean square optimal hedging problem on a trinomial lattice using dynamic programming (see the original literature by Bellman [2], and the recent work by Fedotov and Mikhailov [11, 12] and references therein). We show that one can find the price of a call option under any given probability distribution for the underlying stock return directly by assigning suitable probabilities on the lattice and that these probabilities are assigned based on a network flow optimization which can be solved efficiently by quadratic programming. A computational complexity analysis shows that the number of iterations for dynamic programming and the number of parameters in the network flow optimiza-
tion are both of square order with respect to the number of periods, which indicates that the total computational complexity is highly tractable; this may be a great advantage in using lattice-based computational models. We also illustrate that our methodology generates the implied volatility smile by numerical experiments. Although one may think that our results are related to the so-called implied trees developed in [8, 9, 23] in the sense that the probability assignment on trees is done based on empirical data, an important difference is that our model is based on real probabilities whereas implied binomial trees are based on risk neutral probabilities. Also note that the same technique provided in this paper can be extended to other types of options, including many exotics (such as barriers, compounds, and others) and options with time optionality (see Subsection 3.2).

This paper is organized as follows: We state the problem formulation in Section 2. Section 3 provides the main result and consists of three subsections. We first explain trinomial tree optimal hedging in Subsection 3.1. In Subsection 3.2, we formulate a network flow optimization to assign probabilities on a trinomial lattice, and present an algorithm for pricing a call option on a multi-period lattice. Then we analyze computational complexity of the algorithm in Subsection 3.3. Numerical experiments are made to illustrate our proposed formula in Section 4. Section 5 offers some concluding remarks.

2 Problem Formulation

We consider a market consisting of two basic securities, a risky asset (or stock) and a riskless bond, in the time interval \( t \in [0, t_N] \), where traders are allowed to purchase and sell the two basic securities at discrete times \( t_k = k \tau, k = 0, 1, \ldots, N \). Let \( S_k \) be the price of the stock at \( t = t_k \), where \( S_k \) maps to \( S_{k+1} \) under a certain stochastic process,\(^1\) and let \( B_k \) be the price of the bond with a constant interest rate \( r > 0 \) which satisfies

\[
B_k = RB_{k-1}, \quad k = 1, \ldots, N. \tag{2.1}
\]

where \( R := 1 + r (> 1) \). Also, we define a portfolio to be a couple \((\Delta_k, \theta_k) \in \mathbb{R}^2\) indexed by time \( k = 0 \ldots N \) and let

\[
\Omega_k := \Delta_k S_k + \theta_k B_k, \quad k = 0 \ldots N \tag{2.2}
\]

be the value of the portfolio, where \( \Delta_k \) represents the number of shares of the stock held between \( t \in [t_k, t_{k+1}) \) and \( \theta_k \) represents the number of bonds held by the trader. Assume that the portfolio is self-financing, i.e.,

\[
\Delta_{k-1} S_k + \theta_{k-1} B_k = \Delta_k S_k + \theta_k B_k, \quad \forall k = 1 \ldots N. \tag{2.3}
\]

Finally, let \( C_k, k = 0, 1, \ldots, N \) be the value of a European call option with a strike price \( K \), which pays

\[
C_N = (S_N - K)^+
\]

\(^1\)We will specify a stochastic process for the stock later.
at maturity $t_N$. The objective is to optimally hedge the payoff of the call option through a self-financing trading strategy and determine the value of the call option at $t = 0$ on the stock whose value is $S_0$ at $t = 0$.

If there exists a trading strategy $(\Delta_k, \theta_k) \in \mathbb{R}^2$, $k = 0 \ldots N$ such that the value of the portfolio perfectly replicates the final payoff of the call option, i.e., $\Omega_N = C_N$, the initial value of the call option must be the same as the initial value of the replicating portfolio, or in fact, $C_k = \Omega_k$ must hold for all $k = 0, \ldots, N - 1$ due to an absence of arbitrage argument. This situation occurs when the price of the stock at $t = t_k$, $S_k$, takes two possible states at $t = t_{k+1}$, and Cox, Ross, and Rubinstein [7] showed perfect replication is possible with a self-financing portfolio (this is the well-known binomial lattice option pricing formula). In this formula, the real probability disappears and the unique “risk-neutral probability” dominates the pricing model. On the other hand, if the stock model is governed by a continuous-time geometric Brownian motion process, it is known that there exists an analytic solution for a European call option based on the Black-Scholes option pricing formula. It has been shown [7] that the Black-Scholes formula can also be derived by taking the limit in the binomial lattice option pricing formula under appropriate parameter setting, and that perfect replication is possible with a self-financing portfolio by trading continuously based on dynamic hedging [4]. Therefore we see that markets are complete for both the binomial lattice and the Black-Scholes case, where perfect replication is possible and pricing depends solely on the unique risk neutral probabilities.

However, empirical markets are not complete in general, and perfect replication is never possible. Moreover, empirical stock prices are not necessarily Gaussian and their marginal distributions usually possess heavy tails. These discrepancy may result in differences between the Black-Scholes price and the true option’s price, and appears as the “smile effect” or “implied volatility smile” in real markets. In this paper, we will develop an option pricing formula to take into account heavy tail distributions and incompleteness of markets.

To take “incompleteness” into account, let us consider the case where the price of the stock at $t = t_k$ has more than two possible states to take at $t = t_{k+1}$, say $uS_k$, $mS_k$ and $dS_k$. In this case, perfect replication with a self-financing portfolio is not possible anymore because we have more than two constraints with respect to the states, e.g., $uS_k$, $mS_k$ and $dS_k$, for two variables, $\Delta_k$ and $\theta_k$. As a result, the market becomes incomplete. What we can do instead for the incomplete market models is to solve the mean square optimal hedging problem as follows:

**Mean Square Optimal Hedging (MSOH)**

\[
\begin{align*}
\text{Minimize} & \quad E \left[ (C_N - \Omega_N)^2 \right] S_0 \\
\text{Subject to} & \quad (\Delta_k, \theta_k) \in \mathbb{R}^2 \, \text{satisfying (2.3) for } k = 0, \ldots, N - 1.
\end{align*}
\]

(2.4)

Here the expectation in (2) is now defined over the real probabilities. Once we find the trading strategy to solve the MSOH problem, the initial value of the call option may be chosen as

\[ C_0 = \Omega_0 = \Delta_0 S_0 + \theta_0 B_0. \]

4
An important feature of MSOH for an incomplete market model is that the price of a call option depends on its real probabilities. In this paper, we propose an option pricing methodology which takes advantage of this dependence on real probabilities (especially for heavy tailed distributions), and illustrate that our pricing formula generates an implied volatility smile on heavy tailed distributions. For simplicity, we treat the trinomial tree case as an incomplete market model; however the extension to larger numbers of trees is possible and straightforward.

**Remark 2.1** In this paper, we will associate the optimal initial portfolio value of MSOH with the “price” of the option, and assign $C_0 = 0$. Under this price, the mean of $C_N - \Omega_N$ satisfies $E(C_N - \Omega_N | S_0, \Omega_0) = 0$ [12], and the objective function in MSOH becomes the variance of the wealth balance. Therefore, in this situation, the MSOH problem can be thought of as minimizing the risk in the hedge as measured by the variance subject to a zero mean constraint. Although it has been shown that the above “price” can lead to arbitrage opportunities (see the example by Schweizer [26]), we will continue to refer to it as a price with the possibility of abuse, in keeping with mean-variance theory (see e.g., [18]).

3 Distribution based Option Pricing

3.1 Trinomial Tree Minimum Hedging in Incomplete Markets

To explain the idea, we first consider the optimal hedging problem for the single-period trinomial tree shown in Fig. 1, where $p_i, i = 1, 2, 3$ are given probabilities satisfying $p_1 + p_2 + p_3 = 1$. The three resulting nodes represent multiplication of the stock value by $u, m,$ and $d$, respectively, where $0 < d < m < u$ and $m^2 = ud$. Suppose that the initial price of the stock is $S$ and that $S_N$ takes the value $uS, mS,$ or $dS$ with probability $p_1, p_2$ and $p_3$ at the end of the period. Suppose that the initial value of the risk free asset is given by $B$ with interest rate $R = 1 + r$ ($> 1$). The problem is to minimize

$$E \left[ (C_N - V_N)^2 \right] =$$

$$p_1 (C_u - (\Delta uS + \theta RB))^2 + p_2 (C_m - (\Delta mS + \theta RB))^2 + p_3 (C_d - (\Delta dS + \theta RB))^2 \tag{3.1}$$

over $\Delta$ and $\theta$, for a given $(p_1, p_2, p_3), (u, d, m)$ and strike price $K$, where $C_u, C_m$ and $C_d$ are defined as in Fig. 2, respectively. With these $\Delta$ and $\theta$, the initial price of the call option is obtained as

$$C = \Delta S + \theta B.$$ 

Note that the solution to this minimization problem is computed in closed form [11], i.e., the optimal $\Delta$ and $\theta$ may be calculated directly from given parameters. Therefore, in terms of computational tractability, this option pricing formula is as efficient as the binomial pricing formula for the single-period case.

For the multi-period case, the stock process is described by using a trinomial lattice as shown in the left-hand side of Fig. 3, where the stock price $S_k$ goes either
up ($uS_k$), middle ($mS_k$), or down ($dS_k$) at each step. In this case the solution method can be obtained by working backward based on dynamic programming developed by Bellman [2] (see also a recent work by Fedotov and Mikhailov [11, 12] and references therein). In contrast to the binomial lattice case where the real probabilities are not necessary, we need to specify real probabilities on trees to solve the MSOH problem. This is the essential difference between the binomial and trinomial models and comes from the fact that the market is incomplete, i.e., perfect replication is not possible on a trinomial lattice, which enables us to develop an option pricing formula from the real stock price distribution. In the rest of this paper, we will demonstrate how to assign the probabilities on trees from real data, and illustrate that the formula generates an implied volatility smile for heavy tailed distributions.

3.2 Probability Assignment by Network Flow

To specify the probability at each node, we need a distribution of the log stock return as shown in Fig. 3, where the figure on the right-hand side shows a histogram of the log return for the underlying stock during a specified period (e.g., one year). If we divide by the total number of samples, then the histogram can be considered as
a probability distribution for the log return. Suppose $ud = 1$ and $S_0 = 1$. Then the stock price $S_N$ at the $n$-th node from the top is

$$S_N^{(n)} = u^{N-(n-1)}$$

which is linear in $n$ on a log scale. Since the distribution is a log return on the stock and $S_0 = 1$ (therefore $\ln(S_N)$ is actually a log return), what we do is match the price of the stock with the probability distribution at each node as shown in Fig. 3. Then we obtain the probability to attain $S_N^{(n)}$ at the end.

Let $S_k^{(n)}$, $k \in [0, N]$, $n \in [1, 2k + 1]$ be the price of the stock at the $k$-th period on the $n$-th node from the top, where $S_0^{(1)} = S_0$. If $ud = 1$ and $S_0 = 1$, then $S_k^{(n)}$, $k \in [1, N]$ is given by

$$S_k^{(n)} = u^{k-(n-1)}.$$

Also let

$$P_k^{(n)}, \ k \in [0, N], \ n \in [1, 2k + 1]$$

be the probability to attain $S_k^{(n)}$, where $P_0^{(1)} = 1$ and

$$\sum_{n=1}^{2k+1} P_k^{(n)} = 1, \ \forall k \in [0, N].$$

The price of the stock and the probability to attain that price are arranged on the trinomial lattice as shown in Table 1. From the real distribution, we already know $P_N^{(n)}$, $n \in [1, 2N + 1]$. Let

$$P_k^{(n)}, P_k^{(n)}, P_k^{(n)}, \ k \in [0, N], \ n \in [1, 2N + 1]$$

be probabilities from $S_k^{(n)}$ to $S_{k+1}^{(n)}$, $S_{k+1}^{(n+1)}$, and $S_{k+2}^{(n+2)}$, respectively. These probabilities correspond to an upward, even, and downward move from one node to another on the trinomial tree, e.g., $p_1$, $p_2$ and $p_3$ in Fig. 1. We want to calculate probabilities $P_k^{(n)}$, $P_k^{(n)}$, $P_k^{(n)}$, for all $k$ and $n$ to attain given end point probabilities $P_N^{(n)}$, $n \in [1, 2N + 1]$. These probabilities are obtained by solving a quadratic programming problem as given below.

Consider a network flow with a source $s$ and a sink $s'$ as shown in Fig. 4, where we have eight nodes between the source $s$ and the sink $s'$. We first note that the left hand side of the network consists of a two period trinomial lattice. Each flow travels from the left to the right and finally into the sink $s'$. Let $f_{0,u}^{(1)}$, $f_{0,m}^{(1)}$ and $f_{0,d}^{(1)}$ be the capacities of each edge from the source and let $f_{1,u}^{(n)}$, $f_{1,m}^{(n)}$ and $f_{1,d}^{(n)}$ ($n = 1, 2, 3$) be those from the $n$-th node at the first period as shown in Fig. 4. Also let $\bar{f}^{(1)}$, $\bar{f}^{(2)}$, $\bar{f}^{(3)}$, $\bar{f}^{(4)}$ and $\bar{f}^{(5)}$ be capacities into the sink from the nodes
the end of the lattice as shown in Fig. 4. We assume that the total flow out of the source is one; therefore the total flow into the sink is also one, i.e.,

\[ \bar{f}^{(1)} + \cdots + \bar{f}^{(5)} = 1. \quad (3.2) \]

We are interested in finding admissible capacities \( f_0^{(1)}, f_0^{(1)}, f_0^{(1)}, f_1^{(n)}, f_1^{(n)} \) and \( f_1^{(1)} \) for a given \( f^{(n)} \), satisfying (3.2) for the capacitated network of Fig. 4. The problem can be solved as a quadratic programming problem as follows: Since the total flow is one, we have

\[ f_0^{(1)} + f_0^{(1)} + f_0^{(1)} = 1. \quad (3.3) \]

The flow with capacity \( f_0^{(1)} \) separates to \( f_1^{(1)}, f_1^{(1)}, \) and \( f_1^{(1)} \), respectively, and therefore

\[ f_1^{(1)} + f_1^{(1)} + f_1^{(1)} = f_0^{(1)}. \quad (3.4) \]

Similarly,

\[ f_1^{(2)} + f_1^{(2)} + f_1^{(2)} = f_0^{(n)} \]

\[ f_1^{(3)} + f_1^{(3)} + f_1^{(3)} = f_0^{(n)} \]

Moreover we have five linear constraints with respect to \( f^{(n)}, n \in [1, 5] \):

\[
\begin{align*}
\bar{f}_1 &= f_1^{(1)} \\
\bar{f}_2 &= f_1^{(1)} + f_1^{(2)} \\
\bar{f}_3 &= f_1^{(1)} + f_1^{(2)} + f_1^{(3)} \\
\bar{f}_4 &= f_1^{(3)} + f_1^{(3)} \\
\bar{f}_5 &= f_1^{(3)} + f_1^{(3)}
\end{align*}
\]

Table 1: Stock price and the corresponding probability distribution

| \( S_N^{(1)} \) | \( S_N^{(2)} \) | \( \ldots \) | \( S_N^{(5)} \) | \( P_N^{(1)} \) | \( \ldots \) | \( P_N^{(5)} \) |
| \( S_2^{(1)} \) | \( S_2^{(2)} \) | \( \ldots \) | \( S_2^{(3)} \) | \( P_2^{(1)} \) | \( \ldots \) | \( P_2^{(3)} \) |
| \( S_3^{(1)} \) | \( S_3^{(2)} \) | \( \ldots \) | \( S_3^{(3)} \) | \( P_3^{(1)} \) | \( \ldots \) | \( P_3^{(3)} \) |
| \( S_3^{(2)} \) | \( S_3^{(3)} \) | \( \ldots \) | \( S_3^{(4)} \) | \( P_3^{(2)} \) | \( \ldots \) | \( P_3^{(4)} \) |
| \( S_3^{(3)} \) | \( S_3^{(4)} \) | \( \ldots \) | \( S_3^{(5)} \) | \( P_3^{(3)} \) | \( \ldots \) | \( P_3^{(5)} \) |
| \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( P_N^{(2N+1)} \) |
| \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( \ldots \) | \( P_N^{(2N+1)} \) |
Finally (3.2) is added to these constraints, and we assume that the admissible capacities must be positive. Then, we notice that the problem of finding $f_{1,0}^{(1)} > 0$, $f_{0,1}^{(1)} > 0$, $f_{1,0}^{(n)} > 0$, $f_{1,1}^{(n)} > 0$ and $f_{1,2}^{(n)} > 0$ ($n = 1, 2, 3$) can be solved as a quadratic programming problem with linear constraints (3.2), (3.3), (3.4), (3.5), and (3.6).\(^2\)

![Network flows on trinomial lattice](image)

Fig. 4: Network flows on trinomial lattice

We now assume that $f_i$ in (3.6) are given by the end point probabilities $P_{2,i}^{(1)}$, $i \in [1, 5]$ and that all the capacities are positive. Let

$$
\begin{bmatrix}
P_{0,u}^{(1)} & P_{0,m}^{(1)} & P_{0,d}^{(1)} \\
P_{1,u}^{(n)} & P_{1,m}^{(n)} & P_{1,d}^{(n)}
\end{bmatrix} =
\begin{bmatrix}
f_{0,u}^{(1)} & f_{0,m}^{(1)} & f_{0,d}^{(1)} \\
f_{1,u}^{(n)} & f_{1,m}^{(n)} & f_{1,d}^{(n)}
\end{bmatrix}
$$

where $* = u, m, d$ if $n = 1, 2, 3$, respectively. $P_{0,u}^{(1)}$, $P_{0,m}^{(1)}$, $P_{0,d}^{(1)}$, $P_{1,u}^{(n)}$, $P_{1,m}^{(n)}$, and $P_{1,d}^{(n)}$ ($n = 1, 2, 3$) can be considered as probabilities, because they lie between 0 and 1,\(^3\) and each set of probabilities sums up to one. Moreover these probabilities attain the given end point probabilities $P_{2,i}^{(n)}$, $i \in [1, 5]$ on the lattice. Overall, we see that the problem to find probabilities to attain any given end point distribution can be solved as a quadratic programming problem. Note that this result can easily be generalized to the case with any number of periods and any number of branches (see Appendix A).

\(^2\)A quadratic objective function will be specified in Subsection 3.3.

\(^3\)These probabilities lie between 0 and 1 whenever the admissible capacities in the network flows are positive.
3.3 Algorithm and Computational Complexity Analysis

We are now in a position to present the algorithm to find the initial price of a call option as follows:

Algorithm 1

Step 1): For a given probability distribution for the log stock return over a specified period, choose $u, d, m = \sqrt{ud}$ and the number of periods $N$, and assign the end point probabilities $P^{(n)}_{\infty}$, $n \in [1, 2N + 1]$ as in Fig. 3.

Step 2): For specified end point probabilities $P^{(n)}_{\infty}$, $n \in [1, 2N + 1]$, determine the probabilities on the edges for the $N$ period trinomial lattice by solving the network flow optimization.

Step 3): Compute the initial price of the call option $C_0$ by dynamic programming on the trinomial lattice with probabilities obtained in Step 2).

Note that, in the network flow optimization problem provided in the above algorithm, the admissible capacities satisfying (3.6) are not unique in general, and there may exist infinitely many probabilities which attain the same final distribution. We can reduce this freedom by posing more constraints or constructing an objective function to be minimized. In Appendix B, we construct a quadratic objective function which minimizes the error between the assigned probabilities and a set of target probabilities such that up, middle, and down probabilities at each node are as close as possible to the target probabilities, from the following reasons:

- The problem with the quadratic objective function can still be considered computationally tractable, because it can be solved as a convex quadratic programming problem.

- The stock distributions in empirical markets are usually not very different from Gaussian except for heavy tails, and Gaussian distributions can be approximated very well using a trinomial lattice with the same up, middle, and down probabilities at each node, which also corresponds to an approximation of geometric Brownian motion. Using this objective function allows us to generate such a trinomial lattice process if we assume that the final distribution of the log stock return follows a Gaussian distribution (or a geometric Brownian motion). Therefore the choice of this objective function implies that we want the process on the lattice to be as close as possible to a geometric Brownian motion given a final distribution which is not necessarily Gaussian.

In the next section, we illustrate how our proposed formula works by numerical experiments. We will also confirm that our formula generates the implied volatility smile for heavy tailed distributions.

Next, we will analyze the computational complexity of the proposed algorithm and show that the number of iterations for dynamic programming and the number of parameters in the network flow optimization are both of square order with respect to the number of periods. Note that the dynamic programming in Step 3) consists of a finite number of quadratic minimization problems having a closed form solution.
similar to the one step case. Since larger $N$ gives a more accurate representation for the distribution of the stock at the end of the lattice, we need to consider the relation between the total number of iterations and the number of periods $N$. Let $\Psi(N)$ be the total number of iterations. Then $\Psi(N)$ is given by the total number of nodes from $(N - 1)$-th period to the initial time on lattice, and is calculated as

$$\Psi(N) = \sum_{k=0}^{N-1} (2N + 1) = N^2.$$ \hfill (3.7)

This implies that the number of iterations used in dynamic programming is of square order with respect to the number of periods $N$.

We also need to consider the computational tractability for the quadratic programming problem in Step 2). Let $\Phi(N)$ be the number of variables in the quadratic programming problem (see Appendix A). Since quadratic programming problems can be solved efficiently with a polynomial time algorithm, the problem is considered computationally tractable if the order of $\Phi(N)$ is polynomial with respect to $N$. From the definition of $g_k \in \mathbb{R}^{6k+3}$, we have

$$\Phi(N) = \sum_{k=0}^{N-1} (6k + 3) = 3N^2$$ \hfill (3.8)

which is also of square order with respect to $N$.

**Remark 3.1** Note that the same technique presented here can easily be extended to other types of options, including many exotics (such as barriers, compounds, and others) and options with time optionality (such as Americans and Bermudans), although we have only explained option pricing for a European call option. The dynamic programming algorithm only requires a change in the “boundary condition” corresponding to the appropriate option type. For example, an American call option would require the additional condition that $\bar{C}_n \geq (S_n - K)^+$ (in addition to $\bar{C}_N = (S_N - K)^+$), where $\bar{C}_n$ denotes the value of the option at time $n \in [0, N]$.

## 4 Numerical Experiments

The objective of this section is to show that the solutions obtained from trinomial lattice MSOH generate the implied volatility smile for heavy tail distributions. We also illustrate that our formula provides a good approximation to the Black-Scholes solution in the Gaussian case. The original data for this simulation is given as follows:

- **Opening price $S_0$:** $62$
- **Volatility $\sigma$:** 20%
- **Risk free rate $r_f$:** 10%
- **Expected rate of return $\mu$:** 15%
- **Expiration $T (= t_N)$:** 4 months from now
- **Number of periods $N$:** 32 (8 trades a month)
- **Basic time period $\delta_t$:** 1/96 ($= 1/(12 \times 8)$)
We first solve the mean square optimal hedging problem with the following up, middle, down probabilities and rates,

\[
[p_1, p_2, p_3] := [1/6, 2/3, 1/6]
\]

\[
m := \exp(\nu \delta t), 
\]

\[
u := \mu - \sigma^2/2.
\]

where \(\nu := \mu - \sigma^2/2\). In this case, the trinomial tree approximates the dynamics of the following geometric Brownian motion

\[
dS_t = \mu S_t dt + \sigma S_t dz,
\]

(see Appendix B). The log-stock distribution at the expiration date \(T\) is given by

\[
\ln(S_T) \sim \mathcal{N}\left(\ln(S_0) + \nu T, \sigma \sqrt{T}\right)
\]

and is illustrated in Fig 5. We are interested in the relation between the option prices obtained from trinomial lattice MSOH and the Black-Scholes prices. To see this, we compute the option prices for different values of strike prices \(K\) as shown in Fig. 6, where we also plot the Black-Scholes prices. Since trinomial lattice MSOH results in Black-Scholes prices almost exactly, it appears that there is only one line although there are actually two lines; one for trinomial lattice MSOH and one for the Black-Scholes formula.

Next, we solve the trinomial lattice mean square optimal hedging problem under heavy tail distributions. We generate heavy tail distributions on a trinomial lattice based on a mixed probability distribution of a Gaussian distribution \(p_g(x)\) and a uniform distribution \(p_u(x)\) as

\[
p_m(x) = \alpha p_g(x) + \beta p_u(x), \quad \alpha > 0, \quad \beta > 0
\]

4We have \(2N + 1\) nodes at the end of trinomial lattice.
in an interval $x \in [a, b]$, where $\alpha$ and $\beta$ are weights satisfying $\alpha + \beta = 1$. If $\beta = 0$, $p_m(x)$ is given by a Gaussian distribution $p_g(x)$. Assume that $\ln(S_T)$ follows the mixed distribution $p_m(S_T)$ and that $p_g(S_T)$ is given by the Gaussian distribution in (4.1). We choose the following values for $\beta$,

$$\beta = 0, 0.1, 0.175, 0.25,$$

and the mixed distributions given by these $\beta$’s are plotted in Fig. 7 with respect to $\ln(S_T)$, where the solid line denotes the Gaussian case $\beta = 0$ and tails become fatter as we increase $\beta$. We assign the end point probabilities based on the mixed distributions and solve the network flow optimization problem with a quadratic objective function to determine mid-point probabilities on the trinomial lattice. We then solve the trinomial lattice MSOH problem for different values of strike price $K$ for each probability distribution, and computed implied volatilities. Fig. 8 shows the implied volatilities corresponding to the solutions, where the implied volatilities are plotted vs. the log of strike prices for different values of $\beta$’s. Note that identical line styles are used in Fig. 8 and for the underlying stock distributions. Clearly, we see that fatter tails increase the smile effect. Moreover, we realize that, for the Gaussian case, the implied volatility line stays flat at 20%, which coincides with the Black-Scholes solution.

**Remark 4.1** Finally, we would like to mention that the price obtained from MSOH is not very sensitive to a change of mean return $\mu$ in the stock; For the Gaussian case, we have confirmed the lack of price sensitivity with respect to the mean return by numerical experiments although we omit the details for brevity. This indicates that the information for the empirical mean of the stock does not have to be very accurate. Note that exact estimation of the mean return is difficult in general.
Fig. 7: The mixed distributions: the solid line denotes the Gaussian case $\beta = 0$ and tails become fatter as we increase $\beta$.

Fig. 8: Implied volatilities vs. the log of strike prices for different values of $\beta$'s.
5 Conclusion

In this paper, we have presented a numerical technique for pricing European style call options based on an arbitrarily given probability distribution. We first formulated an optimal hedging problem, and then solved it on a trinomial lattice by using dynamic programming. We showed that the proposed method allows us to find the price of a call option under any given probability distribution for the underlying stock return directly by assigning suitable probabilities on the trinomial lattice. These probabilities are obtained from a network flow optimization which can be solved efficiently by quadratic programming. A computational complexity analysis demonstrated that the number of iterations for dynamic programming and the number of parameters in the network flow optimization are both of square order with respect to the number of periods, which indicates that the total computational complexity is highly tractable. Numerical experiments illustrated that our methodology generates an implied volatility smile on heavy tailed distributions.

References


Appendix

A Probability Assignment for General Case

In this appendix, we generalize the idea in Section 3 to the case with any number of periods. To state the result, we need to define special matrices denoted by $E_k^{(n)}$ for a given $k \in [0, N-1]$ and $n \in [1, 2k+1]$ by

$$E_k^{(n)} = \begin{bmatrix} 0_{3 \times (n-1)} & I_3 & 0_{3 \times (2k-n+1)} \end{bmatrix}^T \in \mathbb{R}^{(2k+3) \times 3} \quad (A1)$$

where $0_{i \times m} \in \mathbb{R}^{i \times m}$ is a matrix of size $i \times m$ whose entries are all zero and $0_{i \times 0}$ signifies that there is no matrix, e.g., $E_0^{(1)}$ denotes $\begin{bmatrix} 0_{3 \times 0} & I_3 & 0_{3 \times 0} \end{bmatrix}^T = I_3$.

Let $f_k^{(n)} \in \mathbb{R}^3, k \in [0, N-1]$ be a vector defined as

$$f_k^{(n)} := \begin{bmatrix} f_{k,u}^{(n)} & f_{k,m}^{(n)} & f_{k,d}^{(n)} \end{bmatrix}^T \in \mathbb{R}^3, \quad n \in [1, 2k+1].$$

Also let the end point probabilities $P_N^{(n)}, n \in [1, 2N+1]$ be given. Then (3.6) for $N = 2$ with $f^{(n)} = P_N^{(n)}$ is now rewritten as

$$\begin{bmatrix} P_N^{(1)} \\ P_N^{(2)} \\ \vdots \\ P_N^{(2N)} \\ P_N^{(2N+1)} \end{bmatrix} = \begin{bmatrix} E_N^{(1)} & E_N^{(2)} & \cdots & E_N^{(2N-1)} \end{bmatrix} \begin{bmatrix} f_N^{(1)} \\ f_N^{(2)} \\ \vdots \\ f_N^{(2N-1)} \end{bmatrix}. \quad (A2)$$

Since the total value of the flow is one, each value of the flow into one node is the probability that the stock price takes that value at that node. Therefore the following conditions hold:

$$\begin{bmatrix} P_k^{(1)} \\ P_k^{(2)} \\ \vdots \\ P_k^{(2k)} \\ P_k^{(2k+1)} \end{bmatrix} = \begin{bmatrix} E_{k-1}^{(1)} & E_{k-1}^{(2)} & \cdots & E_{k-1}^{(2k-1)} \end{bmatrix} \begin{bmatrix} f_{k-1}^{(1)} \\ f_{k-1}^{(2)} \\ \vdots \\ f_{k-1}^{(2k-1)} \end{bmatrix}, \quad k \in [1, N], \quad (A3)$$

where $P_k^{(n)}$ is the probability to attain the value $S_k^{(n)}$ at the $k$-th period and $n$-th node. The left hand side of (A3) is the value of the flow out of each node, and hence leads to another condition:

$$\begin{bmatrix} P_k^{(1)} \\ P_k^{(2)} \\ \vdots \\ P_k^{(2k)} \\ P_k^{(2k+1)} \end{bmatrix} = \begin{bmatrix} 1^T_3 & 1^T_3 & \cdots & 1^T_3 \end{bmatrix} \begin{bmatrix} f_k^{(1)} \\ f_k^{(2)} \\ \vdots \\ f_k^{(2k+1)} \end{bmatrix}, \quad k \in [0, N-1]. \quad (A4)$$
where $1^T_m \in \mathbb{R}^m$ is a vector whose entries are all one, e.g., $1_3 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$. For $k \in [0, N-1]$, let

$$g_k := \begin{bmatrix} (f_k^{(1)})^T & (f_k^{(2)})^T & \cdots & (f_k^{(2k)})^T & (f_k^{(2k+1)})^T \end{bmatrix}^T \in \mathbb{R}^{6k+3}$$

$$\hat{E}_k := \begin{bmatrix} E_k^{(1)} & E_k^{(2)} & \cdots & E_k^{(2k)} & E_k^{(2k+1)} \end{bmatrix} \in \mathbb{R}^{(2k+3) \times (6k+3)}$$

$$\mathbb{I}_k := \text{diag}(1_3^T, \ldots, 1_3^T) \in \mathbb{R}^{(2k+1) \times (6k+3)}$$

$$\mathbb{P}_N := \begin{bmatrix} P_N^{(1)} & P_N^{(2)} & \cdots & P_N^{(2N+1)} \end{bmatrix}^T \in \mathbb{R}^{2N+1}.$$  

By using this notation, we have

$$\hat{E}_{k-1}g_{k-1} = I_k g_k, \quad k \in [1, N-1]$$

$$\mathbb{P}_N = \hat{E}_{N-1}g_{N-1}$$

from (A3), (A4) and (A2). This together with $1_0 g_0 = I_0 g_0 = 1$ yields the following theorem:

**Theorem A.1** Suppose that the end point probability $\mathbb{P}_N$ is given. Then the probabilities on the edges which achieve $\mathbb{P}_N$ at the end of the trinomial lattice are obtained by solving the following linear programming problem:

**Find:** $g_i > 0, \quad i \in [0, N-1]$

**s.t.**

$$\begin{bmatrix} -\mathbb{I}_0 & E_0 & E_1 & -\mathbb{I}_2 & \cdots & -\mathbb{I}_{N-3} & E_{N-3} & -\mathbb{I}_{N-2} & E_{N-2} & -\mathbb{I}_{N-1} & E_{N-1} \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{N-3} \\ g_{N-2} \\ g_{N-1} \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}.$$  

Once $g_i > 0, \quad i \in [0, N-1]$ are found, the probabilities on the edges are given as follows,

$$\begin{bmatrix} p_{0,u}^{(1)} & p_{0,m}^{(1)} & p_{0,d}^{(1)} \\ p_{k,u}^{(n)} & p_{k,m}^{(n)} & p_{k,d}^{(n)} \end{bmatrix} = \begin{bmatrix} f_{0,u}^{(1)} & f_{0,m}^{(1)} & f_{0,d}^{(1)} \\ f_{k,u}^{(n)} & f_{k,m}^{(n)} & f_{k,d}^{(n)} \end{bmatrix} / P_k^{(n)}, \quad k \in [1, N-1], \quad n \in [1, 2k+1],$$

where $P_k^{(n)}$ can be calculated by (A3) or (A4).

### B A Quadratic Objective Function

Here we pose a quadratic optimization to assign up, middle, and down probabilities at each node that are as close as possible to a given set of target probabilities. The idea is to approximate target probabilities at each step such that the differences between the target probabilities and assigned probabilities are minimized. Let
\( \tilde{p}_u, \tilde{p}_m \) and \( \tilde{p}_d \) satisfying \( \tilde{p}_u + \tilde{p}_m + \tilde{p}_d = 1 \) be target probabilities. We would like to minimize the sum of the square error between the target probabilities and assigned probabilities as given by

\[
\sum_{k=1}^{N-1/2k+1} \sum_{n=1}^{2k+1} \left[ (\tilde{p}_u - p_{k,u}^{(n)})^2 + (\tilde{p}_m - p_{k,m}^{(n)})^2 + (\tilde{p}_d - p_{k,d}^{(n)})^2 \right] \tag{A1}
\]

under the flow constraints in (A5). As condition (A1) is not quadratic with respect to the capacities on flows, \( f_{k,u}^{(n)}, f_{k,m}^{(n)} \) and \( f_{k,d}^{(n)} \), we use the following alternative form,

\[
\sum_{k=1}^{N-1/2k+1} \sum_{n=1}^{2k+1} \left[ w_{k,u}^{(n)} (\tilde{f}_{k,u}^{(n)} - f_{k,u}^{(n)})^2 + w_{k,m}^{(n)} (\tilde{f}_{k,m}^{(n)} - f_{k,m}^{(n)})^2 + w_{k,d}^{(n)} (\tilde{f}_{k,d}^{(n)} - f_{k,d}^{(n)})^2 \right],
\tag{A2}
\]

where \( w_{k,u}^{(n)}, w_{k,m}^{(n)} \) and \( w_{k,d}^{(n)} \) are weights and \( \tilde{f}_{k,u}^{(n)}, \tilde{f}_{k,m}^{(n)} \) and \( \tilde{f}_{k,d}^{(n)} \) are target capacities on flows such that

\[
\tilde{p}_u = w_{k,u}^{(n)} \cdot \tilde{f}_{k,u}^{(n)}, \quad \tilde{p}_m = w_{k,m}^{(n)} \cdot \tilde{f}_{k,m}^{(n)}, \quad \tilde{p}_d = w_{k,d}^{(n)} \cdot \tilde{f}_{k,d}^{(n)}.
\]

In the Gaussian case where the log of the stock return follows a normal distribution with mean \( \mu T \) and variance \( \sigma^2 T \), i.e.,

\[
\ln \frac{S_N}{S_0} \sim N(\mu T, \sigma^2 T),
\]

a stock process given by a geometric Brownian motion can be approximated almost exactly by a trinominal lattice stock process by choosing the following parameters for the lattice,

\[
m = \exp (\nu \delta t), \quad u = \exp \left( \nu \delta t + \sigma \sqrt{3\delta t} \right), \quad d = \exp \left( \nu \delta t - \sigma \sqrt{3\delta t} \right),
\]

\[
\tilde{p}_u = 1/6, \quad \tilde{p}_m = 2/3, \quad \tilde{p}_d = 1/6,
\]

where \( \delta t \) is a basic time period. Moreover, based on these target probabilities, we can always find \( w_{k,u}^{(n)}, w_{k,m}^{(n)} \) and \( w_{k,d}^{(n)} \) such that the end point probabilities approximate the Gaussian distribution, and the resulting pricing formula provides a good approximation to the Black-Scholes solution as illustrated in Section 4. Even if the stock distribution is given by a non-Gaussian distribution with heavy tails, we may still use the weights based on the approximated Gaussian distribution, because, in real markets, stock distributions usually approximate Gaussians except in their tails.