

A SMILING TREE

AN EMPIRICAL EVALUATION ON BINOMIAL TREE
METHODS FOR LOCAL VOLATILITY MODEL

By

Mouaz Abdul Ghani Al Alem

A thesis submitted in partial fulfilment of the
requirement for the degree of

MSc in Finance and Banking

Faculty of Business
British University in Dubai

September 2009

Abstract

This study compares between the standard Black-Scholes model and two local volatility models of implied binomial trees for PowerShare index options with regards to the pricing accuracy when evaluated against actual market prices. With Black, F. and M. Scholes (1973): The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, volume: 81, pp. 637 – 59 model as a benchmark, two local volatility models were analyzed: Derman and Kani's [Derman, E., & Kani, I., 1994. The Volatility Smile and Its Implied Tree. *Risk*, 7, 32–39] and Barle and Ckici's [Barle, S and N. Cakici, (1998): How to Grow a Smiling Tree. *The Journal of Financial Engineering*, Vol. 7, No. 2, June 1998]. The model suggested by Barle and Cakici shows the best performance followed by Derman and Kani. Black-Scholes performance on the other hands was significantly lower than the two models. This is attributed to the fact that Black-Scholes model adopts a constant volatility regardless of option's strike price or time to maturity. This finding is consistent at different moneyness levels and for different maturity periods.



DISSERTATION RELEASE FORM

Student Name Mouaz Alem	Student ID 60026	Programme Finance and Banking	Date 30 September 2009
----------------------------	---------------------	----------------------------------	------------------------------

<p>Title</p> <p>A SMILING TREE</p> <p>AN EMPIRICAL EVALUATION ON BINOMIAL TREE METHODS FOR A LOCAL VOLATILITY MODEL</p>

I warrant that the content of this dissertation is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.

I understand that one copy of my dissertation will be deposited in the University Library for permanent retention.

I hereby agree that the material mentioned above for which I am author and copyright holder may be copied and distributed by The British University in Dubai for the purposes of research, private study or education and that The British University in Dubai may recover from purchasers the costs incurred in such copying and distribution, where appropriate.

I understand that The British University in Dubai may make that copy available in digital format if appropriate.

I understand that I may apply to the University to retain the right to withhold or to restrict access to my dissertation for a period which shall not normally exceed four calendar years from the congregation at which the degree is conferred, the length of the period to be specified in the application, together with the precise reasons for making that application.

Signature

A Smiling Tree

Table of Contents

Table of Contents.....	5
Chapter 1. Introduction	6
Chapter 2. Literature Review	10
2.1. Binomial and trinomial Trees models	11
2.2. Volatility	12
2.2.1 Historical Volatility	12
2.2.2 Implied Volatility.....	15
2.2.3 Local Volatility.....	16
2.3. Volatility Smile	16
2.3.1 Concept and Discovery	16
2.3.2 Volatility Smile Example	17
2.4. Modelling the volatility smile	18
2.4.1. Local Volatility Models	19
2.4.2. Stochastic Volatility Models	19
2.4.3. Jump Diffusion Models	21
2.5. Implied Volatility Trees.....	21
2.5.1. Derman and Kani's Implied Tree	22
2.5.2. Barle and Cakici's model.....	26
Chapter 3. Methodology.....	28
3.1. Data Selection	29
3.2. Data Analysis.....	30
Chapter 4. Results	36
4.1. Comprehensive Analysis	37
4.2. Analyzing Results By Moneyness Level	39
4.3. Analyzing Results By Time To Expiry	41
4.4. Analyzing Results By Strike Price.....	43
Chapter 5. Conclusions.....	44
5.1. Suggestions for Future Researches	47
Bibliography	48
Appendix A. Dispersions For Different Moneyness Levels	50
Appendix B. Dispersions At Different Maturities Levels	53
Appendix C. Excel VBA Functions Specifications	57
Appendix D. Excel VBA Functions Code	60

Chapter 1. Introduction

Until 1970s, little was known about options pricing and the way for getting fair valuation for options. In 1973, Fisher Black and Myron Scholes came up with the first systematic approach for pricing options, which they have described in their paper "The Pricing of Options and Corporate Liabilities". The new Black-Scholes method has largely contributed in putting a consistent and easy-to-use way for pricing options. Both buyers and sellers can use the model to give reliable and consistent results. Additionally, it has laid the foundation for a later research by Cox – Ross and Rubinstein published in year 1979 on using binomial trees models for option pricing. Although using binomial and trinomial trees have become a popular way for options pricing, this did not undermine Black-Scholes importance. In fact, Scholes and Merton have both shared Nobel Prize in economics in year 1997 to recognize the significance of their work.

While the main intention of Black-Scholes was finding a suitable way for estimating options prices, traders have used the model in a different way. Instead of starting with historical volatilities of options to get their Black-Scholes prices, traders used the observed market prices and reverse-engineered the model to obtain volatilities implied by actual market prices. This has provided traders with a way to compare options using implied volatilities rather than options prices.

In 1987, major securities markets around the world had crashed including derivatives and options markets, and a big shortfall was discovered in Black-Scholes. The model's shortfall comes from its assumption of having a unique constant volatility for all options related to one underlier regardless of strike price or expiration. While this was generally accepted before the crash, it was proven afterwards that the constant volatility is not accurate, and volatility should depend on both strike price and maturity of options. Consequently, the concept of implied volatility has emerged and transformed the constant volatility of Black-Scholes into a skew shape resembling to a smile and therefore it has been called volatility smile.

To rectify Black-Scholes deficit, many researchers have suggested modifying the model to embrace volatility smile. Others went to the extent of wiping-out the whole model and replacing it with a more advanced model that is sophisticated enough to emulate the smile. The efforts of replacing Black-Scholes were unsuccessful due to model's simplicity in understanding, and efficiency in calculation. Thus econometricians and financial engineers have soon left the idea of replacing the model and turned their efforts into extending it to capture volatility smile. Their attempts have resulted in the innovation of local volatility models; stochastic volatility models and jump diffusion models.

Local volatility models stay near to Black-Scholes as much as possible, and then introduce little modification to reflect the volatility smile. Volatility here is a dependent variable that varies deterministically according to time to expiry and random stock prices. In stochastic volatility models, the volatility of the stock itself is considered an independent random variable whose evolution is correlated with the stock price. Finally, jump diffusion models add jumps and crashes to the standard Black-Scholes prices. They allow stocks to make an arbitrary number of jumps in addition to undergoing diffusions.

This study examines the usage of implied binomial trees as part of local volatility models to enhance the Black-Scholes formula. It includes detailed reviews for Derman and Kani's model and the introduced enhancements by Barle and Cakici to Derman and Kani.

In their researches, neither Derman and Kani nor Barle and Cakici have evaluated their model's stabilities against real market data. Therefore, the value of this study comes from applying real life options data derived from Chicago Board Options Exchange (CBOE) to verify models validity and check its accuracy level in predicting options prices and implied volatilities. Options on PowerShare Exchange Traded Fund are used in this study for testing purposes.

The main objectives of the study are:

- to verify that both Derman and Kani and Barle and Cakici models outperform Black-Scholes model in predicting implied volatility in the selected sample data
- to validate that the suggestions of Barle and Cakici have enhanced the accuracy for Derman and Kani's model
- to ensure that both prior assumptions are consistent at different moneyness levels and for different time to expiration periods

The study uses Visual Basic for Applications functions (VBA) that are specifically developed in Microsoft Excel for the testing purposes. These VBA functions are used to derive options implied volatilities and compute options prices according to Black-Scholes, Derman and Kani, and Barle and Cakici models.

The remaining of the study is organized as per the followings: Section II reviews the literature of options pricing and the main concepts behind it. Section III describes sample data of CBOE index option prices and explains the methodology undertaken in the tests. Section IV analyses the results of applying real market data to models and assesses models accuracy in estimating option prices through local volatility values. The study concludes in Section V.

Chapter 2. Literature Review

This section will review some basic concepts in options valuation related to binomial trees, historical, implied, local volatilities and volatility smile, and then it will go into detailed study of Derman and Kani's model followed by Barle and Cakici's model.

2.1. Binomial and trinomial Trees models

Since Cox, Ross, and Rubinstein published their research in year 1979 for using binomial trees to price options; it has become a very popular method. A diagram of binomial tree represents all possible paths that might be taken by the stock price till the time of expiration of the option.

According to Cox, Ross, and Rubinstein (1979), the binomial tree's model has similar assumptions such as the ones of Black-Scholes. First, the stock price follows a random walk. Second, in each movement, the stock has a certain probability to move up by a certain percentage, and a certain probability to move down by a certain percentage. As the number of steps increases and the time of each step become shorter, the distribution of stock prices takes lognormal shape identical to Black-Scholes model.

Trinomial trees on the other hands can be used as an alternative way to binomial trees. While binomial trees account for only upward and downward movement in each step, trinomial trees take into consideration three different possible outcomes for stock price evolution; upward, middle and downward. The calculations of trinomial trees are analogous to binomial trees. In both cases, the work starts from the end of the tree towards its beginning, and calculates at each step the value for exercising the option and the value for continuing¹.

¹ For more information regarding using trinomial trees in pricing options; refer to Dupire (1994)

2.2. Volatility

The Options Institute (1995) defines the volatility as the “measure of uncertainty about the returns provided by the stock”. Typically, it falls in the range of 15% to 60% per annum, and might reach over 100% per annum for some stocks in certain periods of high fluctuations. Volatility reflects the risk associated with holding or investing in a certain stock or option. Stocks with higher volatilities might bring better returns than stocks with lower volatility in favourable times, or it might be the source of deeper losses in unfavourable periods. On the contrary, stocks with lower volatilities might have lower gains but at the same time lower losses. It is crucial here to mention that the volatility is an indicator to the magnitude of stock price’s movement and not to the direction of that movement.

There are two main types of volatilities related to securities: historical volatility and implied volatility. Additionally, a new type called local volatility has emerged after 1987 markets crash. The following sections will provide brief explanations for the three concepts.

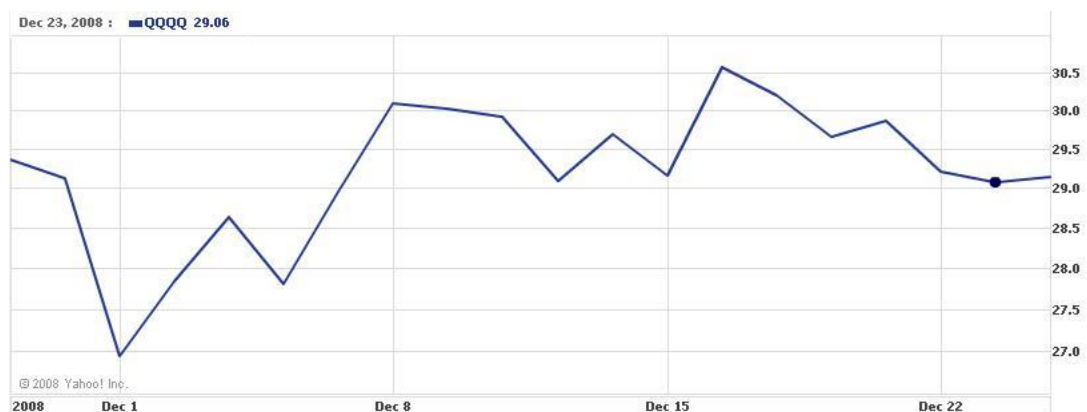
2.2.1 Historical Volatility

Historical volatility is calculated as the average deviation from the average price of a stock during a specific time period. It uses historical data of stock price’s changes in order to predict future stock prices. Also, it assumes that the volatility remains constant. Unfortunately, it is tricky to know how much and how old data should be used. While it is partially true that the more data is used the more volatility is accurate, however, volatility might change over time, and an extreme cautious should be exercised when choosing the time window of the selected data.

Hull (2006, p.286) computes historical volatility based on “standard deviation of the return provided by the stock in one year period when the return is expressed using continuous compounding”, and he advises to use historical data long enough to match the future period investors are willing to invest for.

In the following section, an example of historical volatility for a stock will be demonstrated. The example uses PowerShare, an Exchange Traded Fund (ETF) that replicates NASDAQ 100 Index. PowerShare is traded on NASDAQ and its options are traded on Chicago Board Options Exchange (CBOE) and are among the most active options in the world. Figure 2.1 shows month-to-date stock prices of PowerShare as of 23 December 2008.

Figure 2.1: month-to-date stock prices of PowerShare as of 23 Dec 2008 (Yahoo Finance)



The above figure shows that the price of the stock witnessed a period of high fluctuations in the first week of December and then it went into a more stable period in the second and third week of December. This fluctuation can be measured by looking at the historical volatility of the stock for the same period as in the following table.

Table 2.1: Historical Volatility of PowerShare as of 23 Dec 2008
www.ivolatility.com


	Current	1 WK AGO	1 MO AGO	52 wk Hi/Date	52 wk Low/Date
HISTORICAL VOLATILITY 					
10 days	35.94%	45.87%	62.68%	95.22% - 20-Oct	12.21% - 28-Feb
20 days	49.31%	60.66%	68.13%	82.41% - 07-Nov	16.84% - 30-May
30 days	57.52%	61.37%	76.21%	76.21% - 21-Nov	17.80% - 04-Jun

Table 2.1 shows 10 days, 20 days and 30 days historical volatility values of PowerShare stock as per 23 December 2008. Additionally, the table compares the three volatility values (10 days, 20 days and 30 days) of the stock to its prior values one week ago, one month ago, 52 weeks high and 52 weeks low. These values are exhibited in the following chart.

Figure 2.2: historical volatility comparison chart for PowerShare

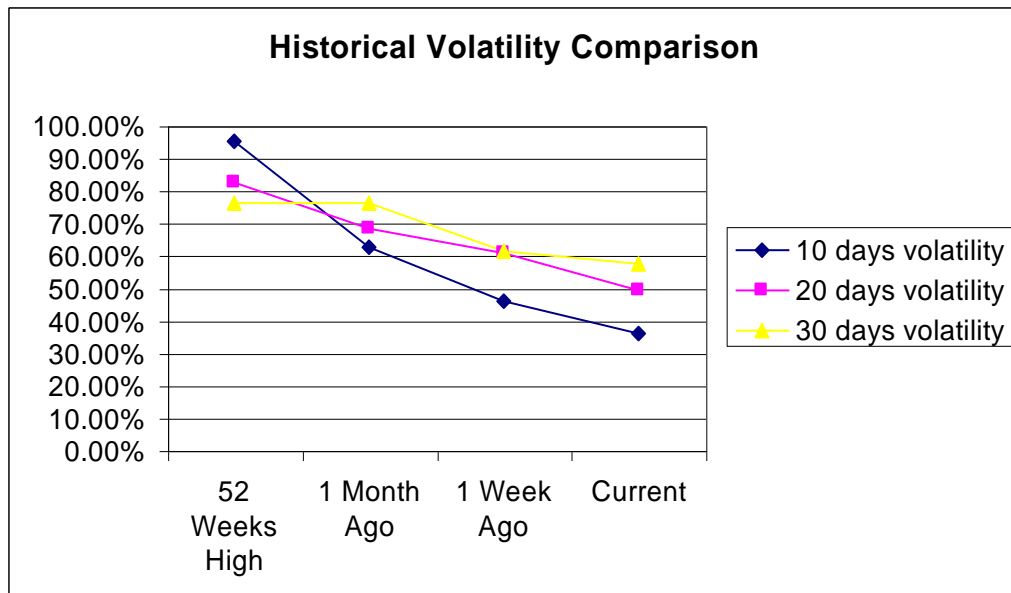


Figure 2.2 presents a chart for rolling historical volatility values presented in table 2.1. The figure shows a declining trend in historical volatilities for PowerShare stock which is consistent with the declining trend in stock fluctuations as in figure 2.1. It is interesting to see that 10 days volatility declines faster than 20 days and 30 days volatility values.

2.2.2 Implied Volatility

Black-Scholes model was originally designed to calculate option's price starting from underlying stock price, exercise price, risk free rate, time to expiration and the volatility of stock. However, the model is used in reality in a different fashion. Traders are using observed market prices of options and reverse-engineer Black-Scholes to find out equivalent volatility values. This is commonly referred to as "implied volatility" of the option as it is implied by actual market prices rather than theoretical values.

Implied volatility reflects investors' future expectations of stock movement instead of what has happened in the past. While historical volatility is affected by the previous performance of the stock, it is the implied volatility that reveals the coming events to the stock such as earnings announcement or mergers and acquisitions events. Implied volatility has become so important that options are quoted in terms of volatility rather than price between professional traders. An example of implied volatility is presented in the table below.

Table 2.2: Implied Volatility of PowerShare as of 23 December 2008
(www.ivolatility.com)

	Current	1 WK AGO	1 MO AGO	52 wk Hi/Date	52 wk Low/Date
IMPLIED VOLATILITY					
IV Index call	41.64%	45.68%	67.74%	79.19% - 20-Nov	18.67% - 30-May
IV Index put	39.94%	48.56%	66.22%	75.77% - 20-Nov	19.32% - 30-May
IV Index mean	40.79%	47.12%	66.98%	77.48% - 20-Nov	18.99% - 30-May

Table 2.2 shows the implied volatility for PowerShare as of 23 December 2008. Like historical volatility, there is a downward trend in implied volatility. For example, the mean implied volatility for PowerShare index recorded 66.98%, 47.12% and 40.79% for one month ago, one week ago and current value respectively on 23 December 2008.

2.2.3 Local Volatility

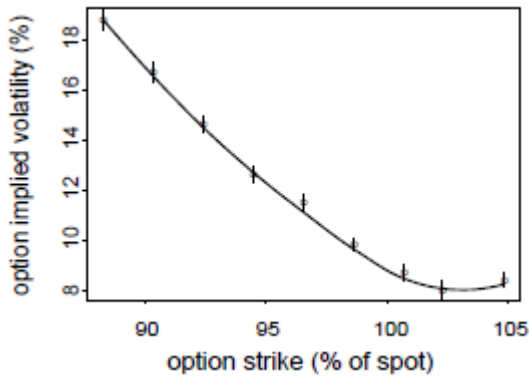
Unlike Black-Scholes model which assumes a constant historical volatility, in local volatility models local volatility represents a function $\sigma(S, t)$ that is dependent on both strike price and time to maturity of the option and is consistent with market prices for all option prices on a given underlier. Detailed discussion on local volatility models comes at a later section.

2.3. Volatility Smile

2.3.1 Concept and Discovery

According to Rubinstein (1994), volatility smile phenomenon was first documented and recognized after security markets crash in 1987. Before the crash, all options on one stock had the same implied volatility regardless of strike price or exercise date. However, since 1987, implied volatility of options on one stock took a U-shape resembling a smile. Deep In-The-Money (ITM) options and deep Out-The-Money (OTM) options have higher implied volatilities than At-The-Money (ATM) options. The closer you get to ATM strikes, implied volatility decreases, and the more you move far-of ATM in both directions implied volatility increases. In recent years, the volatility smile has mostly disappeared from all markets and implied volatility surface has taken the shape of skew. Figure 2.3 demonstrates volatility smile for S&P options market as of Jan 31, 1994.

Figure 2.3: Implied Volatilities of S&P 500 Options on Jan 31, 1994. Derman (1994)



Volatility smile or skew tells that Black-Scholes, with its over-simplicity, is incapable of capturing all the facts regarding options prices. Black-Scholes claims that all options concerning a single stock have the same volatility regardless of its strike or time to expiry. Yet, reality tells us a different story. Reality says that each option has its own implied volatility value depending on strike price; exercise date and the type of the option (call or put) despite the fact that all options are laying on the same underlier.

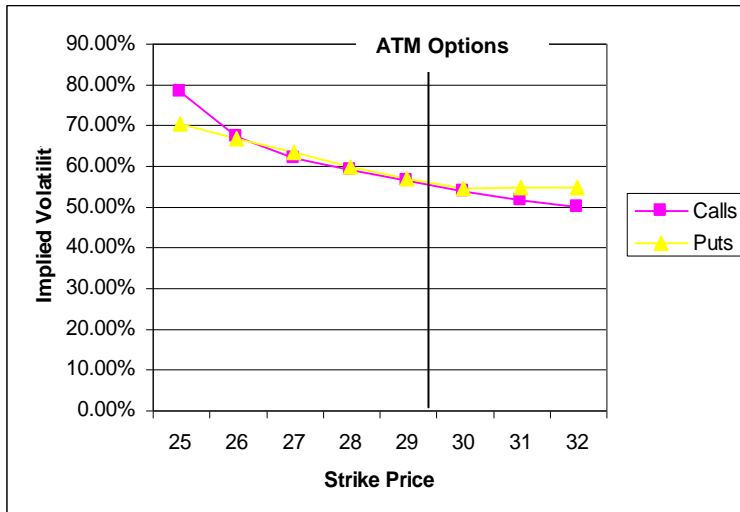
2.3.2 Volatility Smile Example

Table 2.3 and Figure 2.4 show implied volatilities for options on PowerShare expiring on 19 Dec 2008 as of 5 Dec 2008. The values were calculated by reverse-engineering Black-Scholes and finding implied volatility values that make the model's theoretical options' prices equal to market prices. The closing price for PowerShare stock on 5 Dec 2008 was 28.94

Table 2.3: implied volatilities for options on QQQQ expiring on 19 Dec 2008 as of 5 Dec 2008

Strike Price	Implied Volatility	
	Calls	Puts
25	78.00%	70.10%
26	67.00%	66.50%
27	61.60%	63.20%
28	58.80%	59.50%
29	56.20%	56.70%
30	53.50%	54.30%
31	51.30%	54.50%
32	49.60%	54.40%

Figure 2.4: implied volatilities for options on PowerShare stock, expiring on 19 Dec 2008 as of 5 Dec 2008 (Yahoo Finance)



The above figure shows that options with different strikes have different implied volatilities, even though these options are on the same underlier. ITM calls for instance, have higher implied volatilities than ATM calls, and the latter has a higher volatility than OTM calls. Similarly, ITM puts have higher implied volatility than ATM puts, which in turn have higher volatility than OTM puts.

2.4. Modelling the volatility smile

As discussed earlier, the main problem with Black-Scholes model is the assumption of having a constant volatility for all options underlying the same stock. While efforts to replace the model failed; some attempts to enhance the model to embrace the volatility smile were more successful. These attempts can be organized in three groups: local volatility models, stochastic volatility models and jump-diffusion models. The following sections briefly discuss these three groups of models.

2.4.1. Local Volatility Models

These are among the first to model the smile. They try to stay as close to the standard Black-Scholes as possible, and then slightly alter the model to reflect the volatility smile. In its essence, local volatility models generalize the assumption behind Black-Scholes of having lognormal process with constant volatility and therefore, volatility is allowed to vary deterministically as a function of future time and future random stock prices. In local volatility models, the evolution of stock price is provided by the equation:

$$\frac{dS}{S} = \mu(S, t)dt + \sigma(S, t)dZ \quad 2.a$$

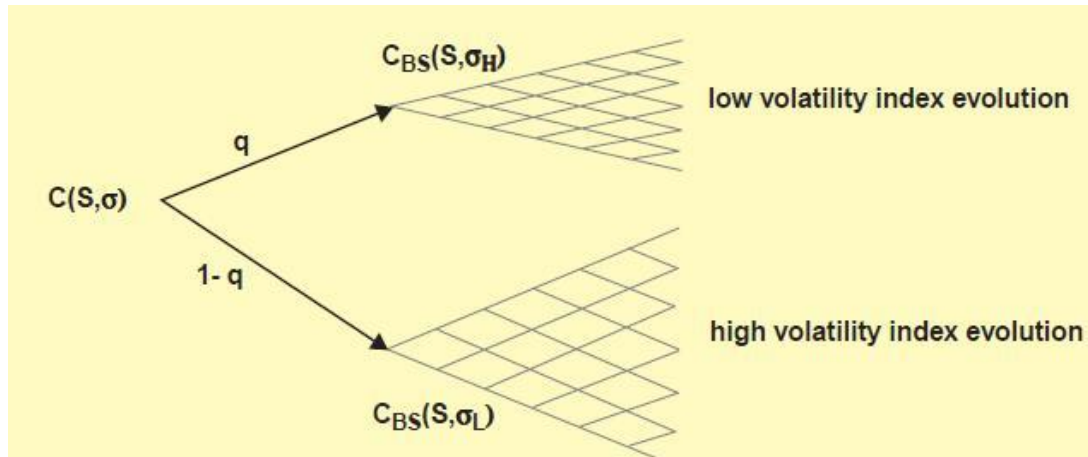
In 2.a, $\sigma(S, t)$ denotes a deterministic function of a stochastic variable S , S denotes underlying stock price and t denotes time to expiry of the option. Section 2.5 will discuss in more details two studies related to local volatility models².

2.4.2. Stochastic Volatility Models

Local volatility models are considered a special case of stochastic volatility models. While the first have variable volatilities based on underlying stock price and time to maturity, the latter considers “the volatility of the stock itself as an independent random variable whose evolution is correlated with that of the stock price”. According to Derman (2003), stochastic volatility models can be thought of as of having two (or more) stock price evolutions, each with its own volatility as in figure 2.5, and option price is the average of Black-Scholes values for high and low volatilities.

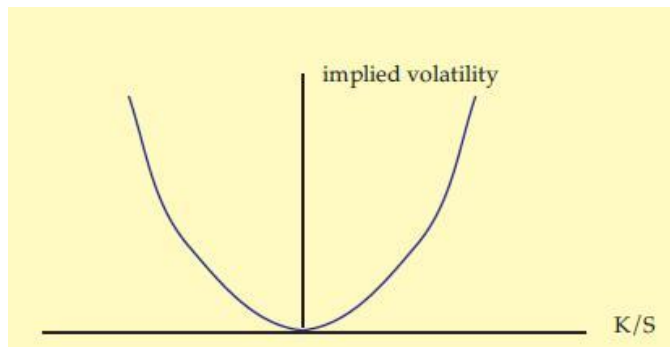
² For more information regarding local volatility models, refer to Derman (Lectures) reference

Figure 2.5: stochastic volatility models. Derman (2003)



When the correlation between underlying stock price and volatility is zero, the result appears as a pure symmetric smile similar to the one found in equal strength currencies. This is demonstrated in figure 2.6.

Figure 2.6: the smile in uncorrelated stochastic volatility models. Derman (2003)



The main feature of stochastic volatility models is that the smile is stable and does not change over time and therefore it is more realistic and resembles real-world smiles. However, as volatility is stochastic, it is not possible to perfectly replicate option's payoff and it is not possible to hedge against it, and therefore the stochastic models are filled up with unknowns and have to go a long way before they become easily usable. A detailed discussion on stochastic volatility is out of the scope of this study³.

³ For more details refer to Derman (2003) and Derman (Lectures)

2.4.3. Jump Diffusion Models

One of the non-realistic assumptions of Black-Scholes is the continuous movement of stock prices. Jump diffusion models have come up to rectify this flaw by adding jumps and crashes to the standard Black-Scholes prices. They allow stocks to make an arbitrary number of jumps in addition to undergoing diffusions. Jump diffusion models realistically take into account jumps such the ones happened in 1987 and 1997. Additionally, they can fit the smile reasonably. Unluckily, these models are not complete in terms of replication, and cannot be hedged with the stock alone or even with an additional option⁴.

2.5. Implied Volatility Trees

According to Barle and Cakici (1998), recombining and constructing binomial and multinomial trees might be one of the simplest, and yet, most initiative and powerful tools available for the valuation of derivative securities. These trees are built from the known prices of European options, and should be consistent with the observed volatility smile. Afterwards, options can be priced in a consistent way with the market. Implied trees should satisfy three criteria:

- it must correctly reproduce the volatility smile
- negative node transition probabilities are not allowed
- the branching process must be risk-neutral at each step

The following sections will study two approaches for constructing implied binomial trees.

⁴ More information is available at Derman (2003) and Derman (Lectures) references

2.5.1. Derman and Kani's Implied Tree

Emanuel Derman and Iraj Kani have published a research in year 1994 with the title "The Volatility Smile and Its Implied Tree". The main objective of their research was to systematically extract from the observed volatility smile a unique binomial tree for the index corresponding to the empirical relationship between stock's volatility and the stock price and maturity date of the option. Derman and Kani (1994) have suggested that from the resulting binomial tree, both the distribution can be calculated as well as the volatility of the index at future times and market levels as implied by option prices. Additionally, the tree can be used for the valuation of other derivatives whose prices are not readily available in the market, such as illiquid European options, American options and exotic options that depend on the details of the index distribution.

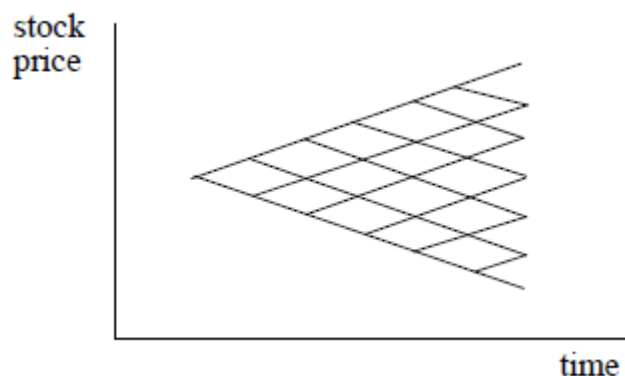
According to Black-Scholes theory, stock prices evolve log-normally with a constant volatility at any time and market level. This evolution can be described by the stochastic differential equation 2.b.

Stochastic differential equation of Black-Scholes

$$\frac{dS}{S} = \mu dt + \sigma dZ \quad 2.b$$

Where S is the stock price, μ is its expected return and dZ is a Wiener process with a mean of zero and variance equal to dt. The implementation of Cox-Ross-Rubinstein binomial tree of equation 2.b is represented in figure 2.7. Notice that stock price evolves with equal logarithmic stock price spacing.

Figure 2.7: Schematic risk-neutral stock tree with constant volatility. Derman (1994)



Unfortunately, the volatility smile that was discovered after market crash in 1987 has caused a flaw in Black-Scholes model and since then, the implied volatility has taken a shape similar to a smile or a skew as explained earlier. It is natural to replace the static volatility in equation 2.b above with a stochastic volatility that is related to both stock price (S) and time to maturity (t) of the option. This is shown in equation 2.c:

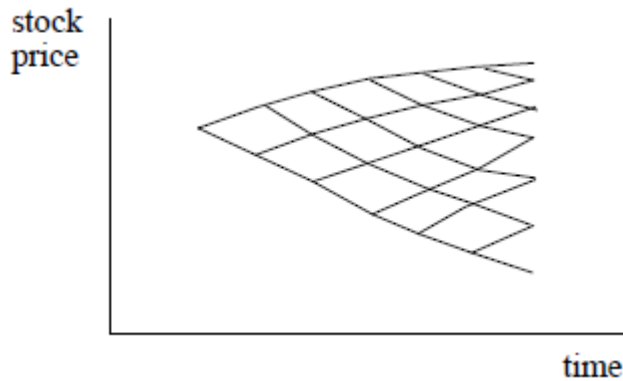
Stochastic differential equation with a volatility function

$$\frac{dS}{S} = \mu dt + \sigma(S, t) dZ \quad 2.c$$

Where $\sigma(S, t)$ represents the local volatility function that depends on both stock price and time to expiry.

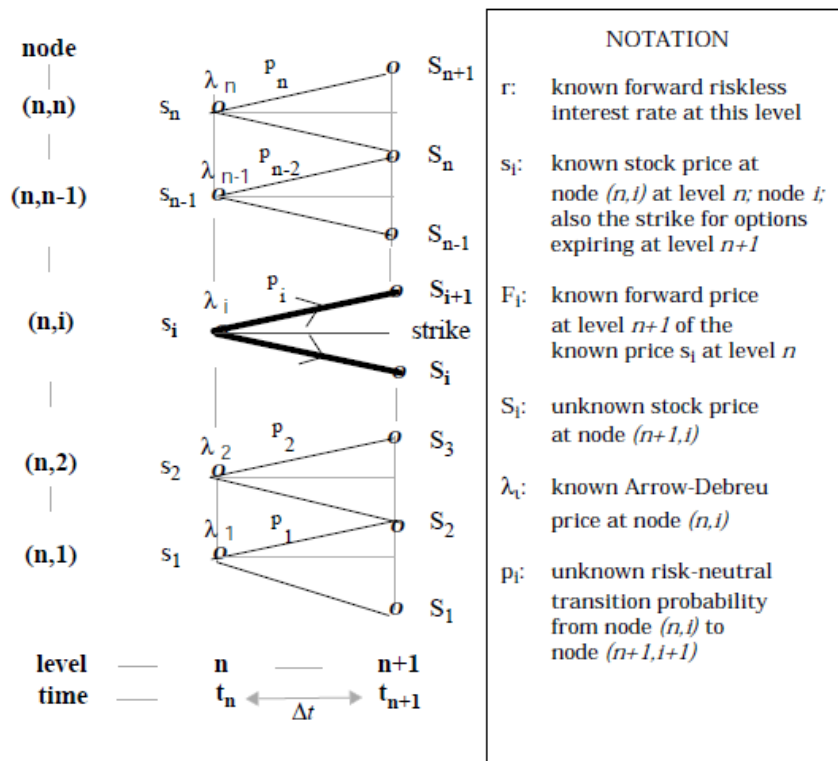
In their paper, Derman and Kani have deduced the $\sigma(S, t)$ numerically from the observed smile rather than involving a special parameter form of the $\sigma(S, t)$ as applied by other researchers. Hence, a more distorted implied tree should replace the standard binomial tree demonstrated in figure 2.7. The resulting tree is illustrated in figure 2.8.

Figure 2.8: The implied tree. Derman (1994)



In the above figure, option prices for all strikes and expirations would decide the position and probability of reaching each node in the implied tree. Let us assume that the first n levels have already been constructed to match the implied volatilities of all options with all strikes out to that time period. Figure 2.9 demonstrates the n^{th} level of the tree at time t_n , with n implied tree nodes and their already known stock prices s_i .

Figure 2.9: constructing the $(n+1)^{\text{th}}$ level of the implied tree. Derman (1994)



Next the tree is used as in figure 2.9 to calculate theoretical values of $2n$ known quantities - the values of n forwards and n options all expiring at time t_{n+1} – and requiring that these theoretical values match market values. Then, depending on the number of nodes, the central node of the standard Cox-Ross-Rubinstein tree can be chosen. In the case number of nodes is odd, the central node's stock price is set to equal today's spot. Otherwise, the average of the natural logarithms of the two central nodes stock prices should be made equal to the logarithm of today's spot price. Finally, the $2n$ equations for the theoretical values of the forwards and the options are derived.

The equation 2.d can be used to get the risk-neutral transition probability from node (n, i) to $(n+1, i+1)$.

Risk neutral transition probability from a lower node to a higher one

$$p_i = \frac{F_i - S_i}{S_{i+1} - S_i} \quad 2.d$$

The initial node's stock price and the higher node's stock price are obtained using equations 2.e and 2.f respectively.

Obtaining upper node stock price

$$S_{i+1} = \frac{S[e^{r\Delta t}C(S, t_{n+1}) + \lambda_i S - \Sigma]}{\lambda_i F_i - e^{r\Delta t}C(S, t_{n+1}) + \Sigma} \quad \text{for } i = n/2 \quad 2.e$$

Obtaining the lower node stock price from a known upper one

$$S_i = \frac{S_{i+1}[e^{r\Delta t}P(S_p, t_{n+1}) - \Sigma] + \lambda_i S_i (F_i - S_{i+1})}{[e^{r\Delta t}P(S_p, t_{n+1}) - \Sigma] + \lambda_i (F_i - S_{i+1})} \quad 2.f$$

These are in brief the steps required to build the binomial tree according to Derman and Kani's model.

2.5.2. Barle and Cakici's model

The main objective of Stanko Barle and Nurset Cakici (1998) paper with the title "How to Grow a Smiling Tree" was to extend Derman and Kani's implied tree, which - according to Barle and Cakici - reproduces the volatility smile accurately in certain circumstances, however it fails in other scenarios such as when the interest rates are high. In addition to solving the problem of high interest rates, Barle and Cakici suggested additional improvements to the implied tree of Derman and Kani.

Barle and Cakici have suggested two major enhancements to Derman and Kani's work. The first major enhancement is related to the way strike price is calculated. Since negative transition probability among tree nodes is forbidden, Derman and Kani uses inequality formula to restrict stock price S_{i+1} as per equation 2.g.

Derman and Kani's inequality equation

$$si \leq Si + 1 \leq si + 1 \quad 2.g$$

However, according to Barle and Cakici, the previous inequality is not necessary true for a given node. The suggested correct inequality formula by Barle and Cakici is 2.h.

Suggested correction to Derman and Kani's inequality equation

$$Fi \leq Si + 1 \leq Fi + 1 \quad 2.h$$

In formulas 2.g and 2.h, F_i , F_{i+1} denotes the forward price; S_{i+1} and S_i are the prices corresponding to the nodes at the $(n+1)$ -th level connected to the node with the price s_i , with K being the strike price. Occasionally, the calculated value of S_{i+1} will fall between s_i and F_i or between s_{i+1} and F_{i+1} . Additionally, Derman and Kani have chosen $K=s_i$ instead of $K=F_i$. This might cause small errors in Derman and Kani's tree, which may accumulate and cause serious discrepancies.

The second major improvement is related to how to choose the centre of the tree. Derman and Kani uses the current stock price S similar to Cox-Ross-Rubinstein binomial tree. Barle and Cakici have suggested allowing the tree to bend along with the interest rate, setting the centre to $Se^{(r-q)t_{n+1}}$. This is to allow the tree to follow the most probable movement of the underlying stock-exponential increase at the risk-free interest rate.

Furthermore, Barle and Cakici have suggested two additional minor enhancements. Sometimes the price S_{i+1} rise outside the interval determined by equation 2.k., leading to a negative transition probability. In such case, Derman and Kani override the price S_{i+1} and set it by hand. The problem is that the overridden price S_{i+1} , may still be violating the inequality as per equation 2.k. Barle and Cakici have suggested choosing any point between F_i and F_{i+1} . i.e: the average of F_i and F_{i+1} to save the tree from negative probabilities. The second minor enhancement proposed is the use of Black-Scholes formula in writing any serious computer program for the implied tree, rather than using Cox-Ross-Rubinstein tree for calculations purposes. Black-Scholes model is much faster, more accurate and requires very light computing power when compared to Cox-Ross-Rubinstein binomial trees.

Chapter 3. Methodology

3.1. Data Selection

Options prices of PowerShares are employed to carry out the testing part of this study. Daily prices (bid / ask price midpoints) of PowerShare options are used for the period 11 February 2002 till 29 May 2002. Data was obtained from web site <http://www.historicaloptiondata.com/> which provides historical daily prices for all options traded on Chicago Board Options Exchange (CBOE). However, the following data are excluded:

- None-liquid PowerShare options that have a volume of less than 10 trades per day are excluded.
- Deep-in-the-money and deep-out-of-the-money options are excluded as well, as these options have little trading activities. Deep-in-the-money options are defined as those options that are less than 75% of ATM options prices, and deep out-of-the-money options are those options that are more than 125% of ATM options prices
- Finally, options that expire on the same date are excluded

For historical volatility calculation, historical stock prices of PowerShare were obtained from Yahoo Finance for the period 2 April 2001 till 10 February 2002. It is assumed that the total trading days are 252 days per year. The calculated historical volatility is used as the constant volatility of Black-Scholes and it equals to 48% throughout the testing period.

The risk free rate used within this study is the three-month treasury bills rate with constant maturities which was obtained from Federal Reserve web site. Since risk free rate has little effect on the results, 11 February 2002 rate of 1.75% was used and it was assumed that it remains constant throughout the testing period for all maturities. Additionally, it was assumed that PowerShare stocks pay no dividend in the sample period. Finally, Derman & Kani and Barle and Cakici models were run for five steps throughout this study.

3.2. Data Analysis

In the remaining of this study, Black-Scholes, Derman and Kani, and Barle and Cakici models will be referred to as BS, DK and BC respectively. Given the described PowerShare options data, the study will estimate how well BS, DK and BC models predict options prices. This is done through measuring the mean difference (absolute mean difference) of models local volatilities and the volatilities implied by market prices of the options. The mean difference was selected because it measures the dispersion or the spread out of two independent variables or series, and it is used here as an indicator for the dispersion of BS, DK and BC models' local volatilities from market implied volatility.

A Lower dispersion or a lower mean difference indicates less difference between predicted local volatility and actual market volatility, and hence the model is performing better in predicting market values. On the contrary, a high dispersion or a high mean difference signifies a bigger difference between predicted local volatility and actual market volatility, and hence the model is under-performing in predicting market values.

One of the major issues during the study was the selection of the local volatility function that will be used to fit the smile. In order to keep the study simple, the volatility is assumed to be dependent on strike price solely and it is derived from Hardle (2002, p.156), and then tuned to fit observed market prices for the first two weeks in the data sample. Equation 3.a represents the volatility function that is used for the data analysis.

$$\sigma_{imp} = 0.4 + \left(\frac{S - K}{S} \right) * 4 \quad \mathbf{3.a}$$

In the previous equation, S refers to stock price and K refers to strike price. This function is used to evaluate the prediction power of both DK and BC models and then to compare it with the standard BS model.

For the evaluation process, seven Visual Basic for Applications functions (VBA) for Microsoft Excel was developed to automate the calculations of Black-Scholes, Cox-Ross-Rubinstein, Derman & Kani and Barle and Cakici models. These functions are listed in table 3.1 with brief explanation regarding each function.

Table 3.1: Excel VBA Function used in analysis.

ID	Function Name	Usage
1	DK	Calculates option price according to Derman and Kani model
2	BC	Calculates option price as per Barle and Cakici model
3	BS_Call	Calculates call option price according to Black and Scholes
4	BS_Put	Calculates put option price according to Black and Scholes
5	CRR	Calculates option price according to Cox-Ross-Rubinstein (CRR)
6	BS_IV	Calculates implied volatility from option price
7	Vol_Smile	Calculates options volatility according to volatility function as in 3.e function

The Excel VBA functions used in this study were mostly derived from Rebel (2004) project and from XploRe Software handbook available online. The code has to undergo many modifications to fit the purposes of study. Full specification of these functions is listed in Appendix A. Actual code comes in Appendix B.

PowerShare options data covers a period of 16 weeks. For testing purposes, the data were divided into eight bands, each band spans a period of two weeks. The details of the resulting bands are listed in table 3.2.

Table 3.2: shows how data was divided into eight bands and details of each band

Band ID (B)	Weeks	Start Date	End Date	Number of Options (N)
1	Week 1 + 2	11 Feb 2002	22 Feb 2002	855
2	Week 3 + 4	25 Feb 2002	8 Mar 2002	1232
3	Week 5 + 6	11 Mar 2002	22 Mar 2002	1253
4	Week 7 + 8	25 Mar 2002	5 Apr 2002	1061
5	Week 9 + 10	8 Apr 2002	19 Apr 2002	1115
6	Week 11 + 12	22 Apr 2002	3 May 2002	1270
7	Week 13 + 14	6 May 2002	17 May 2002	1320
8	Week 15 + 16	20 May 2002	31 May 2002	855
			Total	8961

The detailed steps used in analysing data are as per the followings:

Step 1: Data Organizing

Table 3.3 shows a snapshot of PowerShare data that were obtained from Chicago Board Options Exchange (CBOE) and used for calculations. The data were organized in one Excel sheet to facilitate the calculations and results collection.

Table 3.3: a snapshot of PowerShare data used in the calculations

	A	B	C	D	E	F	G	H	I	J	K
1	Period	Underlier Price	Strike	Type	Data Date	Expiry Date	Maturity (days)	Maturity (years)	IV	Option Price	BS Volatility
2	Week 02	36.63	33	call	2/11/02	2/15/02	4	0.011	0.71602	3.7	0.48
3	Week 02	36.63	34	call	2/11/02	2/15/02	4	0.011	0.550843	2.7	0.48
4	Week 02	36.63	35	call	2/11/02	2/15/02	4	0.011	0.511552	1.8	0.48
5	Week 02	36.63	36	call	2/11/02	2/15/02	4	0.011	0.495912	1.05	0.48
6	Week 02	36.63	37	call	2/11/02	2/15/02	4	0.011	0.474931	0.5	0.48
7	Week 02	36.63	38	call	2/11/02	2/15/02	4	0.011	0.473596	0.2	0.48
8	Week 02	36.63	29	call	2/11/02	3/15/02	32	0.089	0.520707	7.8	0.48
9	Week 02	36.63	30	call	2/11/02	3/15/02	32	0.089	0.526811	6.9	0.48
10	Week 02	36.63	31	call	2/11/02	3/15/02	32	0.089	0.51117	6	0.48
11	Week 02	36.63	32	call	2/11/02	3/15/02	32	0.089	0.482083	5.1	0.48
12	Week 02	36.63	33	call	2/11/02	3/15/02	32	0.089	0.476361	4.3	0.48
13	Week 02	36.63	34	call	2/11/02	3/15/02	32	0.089	0.452949	3.5	0.48
14	Week 02	36.63	35	call	2/11/02	3/15/02	32	0.089	0.440789	2.8	0.48
15	Week 02	36.63	36	call	2/11/02	3/15/02	32	0.089	0.422622	2.15	0.48

Step 2: Calculating Options Prices using BS, DK and BC Models

In the second step, options prices are calculated according to BS, DK and BC models as explained earlier in sections 2.5.1 and 2.5.2. Excel VBA code is used here to automate the calculations of these models as detailed in Appendixes C. and D. Table 3.4 below shows a sample of the calculated prices.

Table 3.4: Calculating BS, DK and BC models prices

	A	B	Q	R	S	T
1	Period	Underlier Price	BS Price	DK Price	BC Price	Market Price
2	Week 02	36.63	3.649	3.636	3.636	3.700
3	Week 02	36.63	2.692	2.663	2.663	2.700
4	Week 02	36.63	1.818	1.764	1.751	1.800
5	Week 02	36.63	1.095	0.967	0.935	1.050
6	Week 02	36.63	0.576	0.465	0.435	0.500
7	Week 02	36.63	0.260	0.174	0.158	0.200
8	Week 02	36.63	7.773	7.742	7.744	7.800
9	Week 02	36.63	6.848	6.775	6.775	6.900
10	Week 02	36.63	5.961	5.847	5.819	6.000
11	Week 02	36.63	5.123	5.034	4.995	5.100
12	Week 02	36.63	4.344	4.220	4.172	4.300
13	Week 02	36.63	3.632	3.407	3.349	3.500
14	Week 02	36.63	2.993	2.680	2.582	2.800
15	Week 02	36.63	2.431	2.175	2.083	2.150

Step 3: Calculating Implied Volatilities for DK and BC models

Rather than comparing prices, models implied volatilities were used to compare and evaluate models correctness and preciseness. In the third step we used models prices calculated in step 2 as input and reversed engineered the standard Black-Scholes model to calculate their implied volatilities. A sample of the results appears in table 3.5 below. Note that historical volatility of 48% was used for Black-Scholes.

Table 3.5: shows an extraction of data sheet used for calculation

	A	B	U	V	W	X
1	Period	Underlier Price	BS IV	DK IV	BC IV	MKT IV
2	Week 02	36.63	0.480	0.136	0.133	0.716
3	Week 02	36.63	0.480	0.411	0.411	0.551
4	Week 02	36.63	0.480	0.423	0.408	0.512
5	Week 02	36.63	0.480	0.390	0.366	0.496
6	Week 02	36.63	0.480	0.407	0.387	0.475
7	Week 02	36.63	0.480	0.405	0.391	0.474
8	Week 02	36.63	0.480	0.446	0.448	0.521
9	Week 02	36.63	0.480	0.423	0.423	0.527
10	Week 02	36.63	0.480	0.417	0.398	0.511
11	Week 02	36.63	0.480	0.444	0.428	0.482
12	Week 02	36.63	0.480	0.440	0.423	0.476
13	Week 02	36.63	0.480	0.417	0.400	0.453
14	Week 02	36.63	0.480	0.401	0.376	0.441
15	Week 02	36.63	0.480	0.420	0.398	0.423

Step 4: Calculating Mean Difference (Absolute Mean Difference)

Models implied volatilities calculated in step 3 are then subtracted from market implied volatility to compute models error in predicting market implied volatility. This is done by taking the absolute value of the difference between models and market implied volatilities and then taking the average across each band (two weeks period) and finally across all eight bands in the data sample. A screen snapshot is shown in table 3.6.

Table 3.6: shows an extraction of data sheet used for calculation

	A	B	AB	AC	AD
1	Period	Underlier Price	ABS (BS IV - MKT IV)	ABS (DK IV - MKT IV)	ABS (BC IV - MKT IV)
2	Week 02	36.63	0.236	0.580	0.583
3	Week 02	36.63	0.071	0.140	0.139
4	Week 02	36.63	0.032	0.088	0.103
5	Week 02	36.63	0.016	0.106	0.130
6	Week 02	36.63	0.005	0.068	0.088
7	Week 02	36.63	0.006	0.068	0.083
8	Week 02	36.63	0.041	0.074	0.073
9	Week 02	36.63	0.047	0.103	0.103
10	Week 02	36.63	0.031	0.095	0.113
11	Week 02	36.63	0.002	0.038	0.054
12	Week 02	36.63	0.004	0.037	0.053
13	Week 02	36.63	0.027	0.036	0.053
14	Week 02	36.63	0.039	0.039	0.064
15	Week 02	36.63	0.057	0.003	0.024

Equation 3.b below is used to calculate the mean difference in the previous step.

$$\frac{\sum_{1}^N |\sigma_{models} - \sigma_{imp}|}{N} \quad \mathbf{3.b}$$

In the previous equation σ_{imp} refers to market implied volatility and σ_{models} denotes volatilities of BS, DK and BC models implied by calculated theoretical prices as in step two. Finally, the symbol N refers to PowerShare option's number across the specific band. For instance, band 1 contains 855 options; band 2 contains 1232 options and so forth.

Chapter 4. Results

4.1. Comprehensive Analysis

First, the study will examine the mean difference of options local volatilities of BS, DK and BC models and the volatility implied by market throughout our testing period. Figures 4.1 and 4.2 illustrate BS, DK and BC local volatilities dispersions from implied market volatility for each time band within our study.

Figure 4.1: Call Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	12.99%	7.56%	7.26%
2	10.31%	4.80%	3.88%
3	14.46%	7.28%	5.79%
4	13.58%	5.60%	3.95%
5	14.33%	7.27%	5.90%
6	12.35%	4.60%	3.26%
7	11.80%	7.00%	6.66%
8	10.48%	2.62%	1.71%
Average	12.54%	5.84%	4.80%

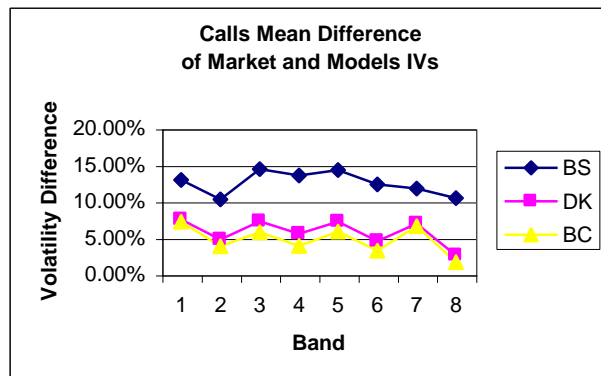
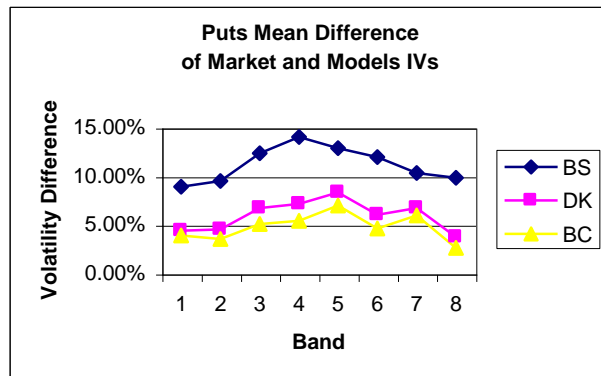


Figure 4.2: Put Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	8.93%	4.42%	3.93%
2	9.52%	4.55%	3.56%
3	12.38%	6.75%	5.11%
4	14.05%	7.16%	5.43%
5	12.91%	8.35%	7.01%
6	12.00%	6.05%	4.64%
7	10.34%	6.74%	6.00%
8	9.86%	3.77%	2.65%
Average	11.25%	5.97%	4.79%



It is clear in figures 4.1 and 4.2 that both DK and BC models have significantly outperformed BS model in predicting market prices for both call and put options by scoring lower in the mean difference test. Additionally, BC model has performed slightly but consistently better than DK across all bands as well.

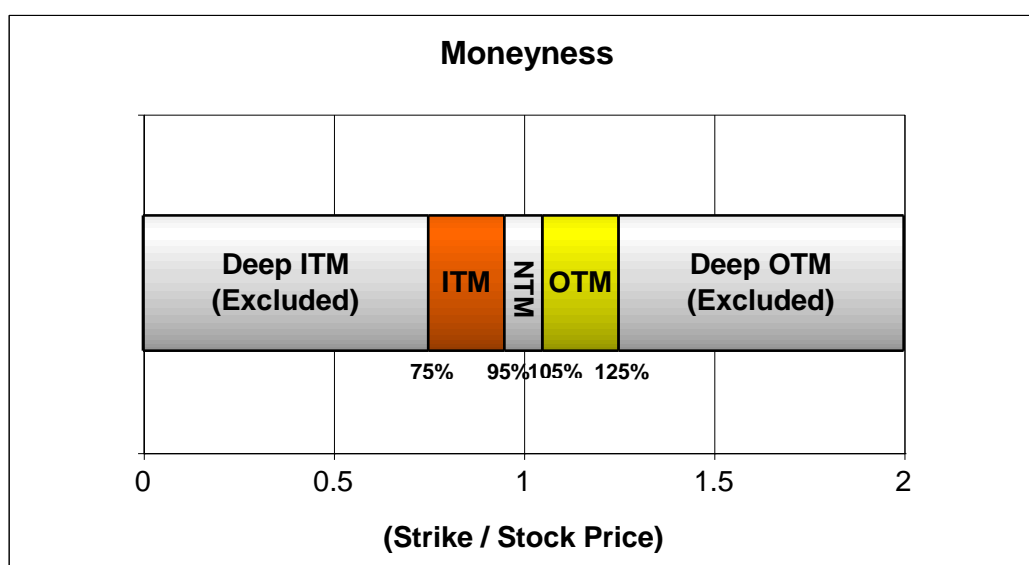
The mean difference of BS constant historical volatility of 48% and implied market volatility in the testing period was within the range of 10.31% to 14.46% with an average of 12.54% for call options. The same range was 8.93% to 14.05% with the average of 11.25% for put options. In DK's model, the mean difference of the model's local volatility and implied market volatility was within the range of 2.62% to 7.56% with an average of 5.84% for call options. The range was 3.77% to 8.35% with the average of 5.97% for put options. The BC's model was the best and the lowest in the prediction as the mean difference of its local volatility and the volatility implied by the market was within the range of 1.71% to 7.26%, with an average of 4.8% for call options and in the range of 2.65% to 7.01% with the average of 4.79% for put options.

The dispersion of BC model's local volatility and market implied volatility was 1% lower than the dispersion of DK's local volatility and market implied volatility. This is consistent with the theory, as the enhancements suggested by BC model have brought better results to DK.

4.2. Analyzing Results By Moneyness Level

In the second test, data is tested at different moneyness levels. As mentioned earlier, deep-in-the-money and deep-out-of-the-money options were excluded and only options within the range of +/- 25% of ATM are included. Based on moneyness level, the data were divided into three categories, ITM options that are less than 5% than ATM, OTM options that are more than 5% than ATM and NTM options in between. The moneyness levels are illustrated in figure 4.3.

Figure 4.3: Moneyness Levels



Appendix A. lists detailed tables and charts of mean differences for models local volatilities and market-implied volatility of both call and put options for ITM, NTM and OTM moneyness levels. Table 4.1 provides summary for these observations.

Table 4.1: DK and BC models average mean difference of local volatilities and market implied volatility

Moneyness	BS		DK		BC	
	Call	Put	Call	Put	Call	Put
ITM	11.57%	14.39%	9.67%	7.04%	8.72%	6.08
NTM	11.08%	11.76%	5.49%	6.21%	4.04%	4.71
OTM	13.97%	8.41%	3.29%	4.96%	2.43%	3.08

Table 4.1 shows that both DK and BC models have considerably outperformed BS model in predicting options prices and market implied volatility at the three moneyness levels for both call and put options. Additionally, BC model has also slightly outperformed DK model in the prediction at all moneyness levels for both call and put options. It was also interesting to see that both DK and BC models have performed the best at OTM calls and puts. The dispersion of DK and BC Models from the market at NTM calls and puts came higher than OTM but still lower than ITM options. ITM options dispersion for both models was the highest and at the same time the least accurate among moneyness levels; although it is still lower than BS dispersion.

For example, the dispersion of DK model was 9.67%, 5.49% and 3.29% for ITM, NTM and OTM call options respectively. In the same manner, the dispersion of BC local volatility's and implied market volatility was 8.72%, 4.04% and 2.34% for ITM, NTM and OTM call options respectively.

4.3. Analyzing Results By Time To Expiry

Next, data will be tested at three maturity levels: short maturity for options with expiry less than 30 days, medium maturity for options expiring within 30 to 90 days and long maturity for options expiring after 90 days.

Appendix B. lists detailed tables and charts of mean differences for models local volatilities and market-implied volatility of both call and put options for long, medium and short maturities as per previous maturity definitions. Table 4.2 provides a summary for these observations.

Table 4.2: DK and BC models average MD local volatilities and market implied volatility

Maturity	BS		DK		BC	
	Call	Put	Call	Put	Call	Put
More than 90 days	12.28%	12.65%	4.01%	5.79%	2.49%	4.00
30 – 90 days	11.02%	9.98%	3.50%	3.77%	2.75%	2.96%
Less than 30 days	17.74%	10.54%	18.42%	12.74%	18.31%	12.50%
11 – 29 days	8.88%	9.61%	6.77%	7.88%	6.21%	7.46%

Table 4.2 shows that dispersions of both DK and BC local volatilities from market implied volatility were significantly lower than BS dispersion at long and medium maturities for both call and put options. Additionally, the dispersion of BC model was slightly lower than DK’s dispersion for long and medium maturity levels for both call and put options. However, it is noticeable that the dispersion of BS from market came lower than DK and BC dispersions for shorter maturities in both call and put options.

The results for long and medium maturities were not conclusive. DK model has performed the best in medium maturities, followed by long maturities for both call and put options. DK dispersion of local volatilities from market implied volatility came as high as 18.42% and 12.74% for call and put options respectively.

The lowest dispersion of BC model was in long maturity calls and medium maturity puts. This is followed by medium maturity calls and long maturity puts. Short maturities registered the highest dispersion in BC model as it scored 18.31% and 12.50% for call and put options respectively.

Apparently, there is a wide dispersion in both DK and BC local volatilities compared to market implied volatility for options with less than 30 days to expiry. This large dispersion has provoked us to make another investigation about its cause. In order to track the reason behind this large dispersion for shorter maturities, an additional test is performed for options with maturities between 11 and 29 days to expire after excluding options with maturities between one and 10 days. The result is illustrated in Appendix B. and is summarized in Table 4.2.

After excluding options that matures within 10 days of the data date, the average dispersion has been significantly reduced from the range of 12.50% - 18.42% to the range of 6.21% - 7.88%. This result goes well along with theoretical studies, and both DK and BC models dispersions have returned to their previous performance that is significantly lower than BS. This behaviour is explained by Dumas (1995), as options with little time to expiry “have relatively small time premia, hence the estimation of volatility is extremely sensitive to no-synchronous option prices and other possible measurement errors”.

Based on the previous analysis, it can be generalized that both DK and BC options provide lowest dispersion in options maturing within 30 – 90 days, with the exception of BC’s model prediction of call options. The next lowest dispersion is for options with maturities longer than 90 days, followed by maturities of 11 to 29 days. Models’ dispersion of options with expiry of 10 days or less was large and inaccurate.

4.4. Analyzing Results By Strike Price

Finally, local volatilities of BS, DK and BC shall be examined and compared to market implied market for call options with 42 days to expiry as of 8 March 2002. The result is illustrated in figure 4.4.

Figure 4.4: BS, DK, BC local volatilities compared to market IV for calls maturing after 42 days as of 8 March 2002

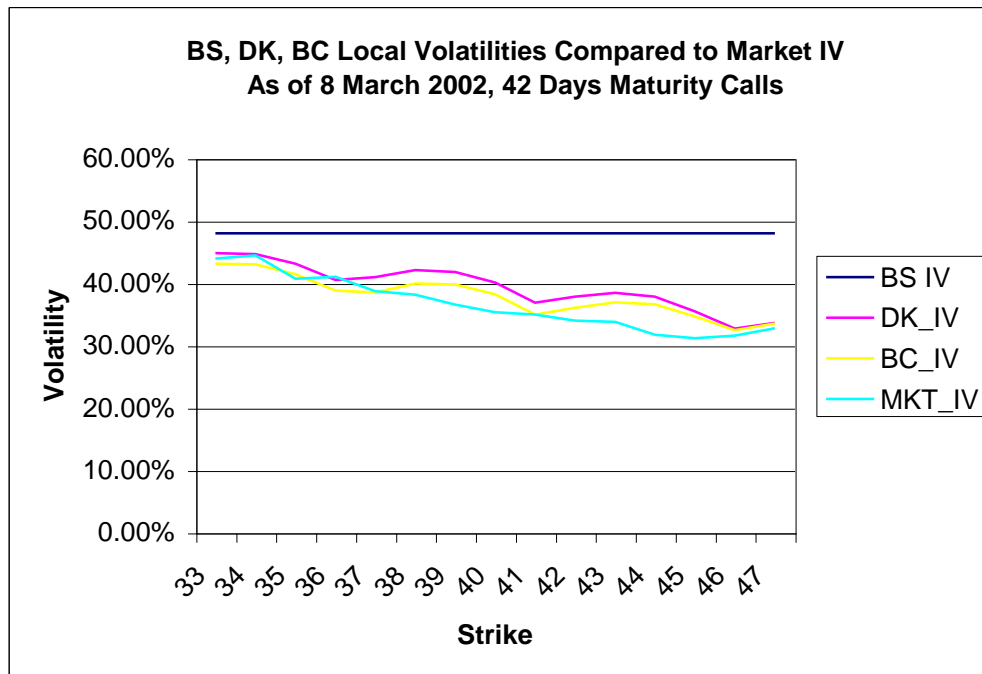


Figure 4.4 demonstrates that local volatilities of DK and BC models were closer to market implied volatility than BS model, and that BC's local volatility was the closest to market implied volatility at most strike prices, and especially with out-of-the-money strikes higher than PowerShare asset price of 38.67. This is consistent with our previous finding that BC model performs the best in OTM options, followed by NTM and ITM options respectively.

Chapter 5. Conclusions

Since its first discovery after stock markets crash in 1987, volatility smile has invaded all options markets worldwide and has entered into all types of options markets from stock options to foreign exchange options. It was soon realized after the 1987 crash, that the Black-Scholes model is not perfect and holds a significant deficiency by assuming that stock volatility is constant regardless of time to maturity or strike price. The crash-phobia sentiment of traders has led them to use options as an insurance policy against similar crashes and hence traders have become ready to pay higher prices to hedge their portfolio using options.

Financial engineers, econometricians and risk officers have tried to come up with a replacement to Black-Scholes that fixes its big shortfall. However, they have recognized that Black-Scholes has its own unique advantages, which made it almost impossible to entirely replace the model. Therefore, replacing the model was substituted by attempts to enhance the model for capturing the smile. Three approaches for improving Black-Scholes model were introduced, local volatility models, stochastic volatility models and jump diffusion models. Each of these approaches has its advantages and disadvantages.

This study has examined Derman and Kani's model and its suggested enhancements by Barle and Cakici. These two approaches are considered part of local volatility models, and they both use Cox-Ross-Rubinstein method of binomial trees with some modifications to capture volatility smile. The study used PowerShare Exchange Traded Fund (ETF) options to test both Derman and Kani and Barle and Cakici models. Mean difference of models predicted local volatilities and implied market volatilities were used as indicator for models performance.

The results of tests came consistent with theoretical studies for both the Derman and Kani's and Barle and Cakici's models. First, Derman and Kani and Barle and Cakici models have significantly and consistently outperformed Black-Scholes model. This is expected as both models adopt a function dependent on asset price in determining option's volatility. Black-Scholes model on the other hands adopt a constant volatility regardless of strike price or time to expiry. Second, Barle and Cakici's model has slightly and consistently outperformed Derman and Kani's model. This is the result of the enhancements suggested by Barle and Cakici. Third, the three models were tested at three different moneyness levels: out-of-the-money, near-the-money and in-the-money. Moreover, the models were tested at three maturity levels: 11 to 29 days, 30 to 90 days and more than 90 days of maturity.

Both Derman and Kani and Barle and Cakici models have outperformed Black-Scholes model at all levels of moneyness and maturities. Additionally, Barle and Cakici's model has slightly outperformed Derman and Kani at all moneyness and maturity levels as well.

Furthermore, the study has showed that the best performance for implied trees models according to their moneyness level was in out-the-money options, followed by near-the-money and in-the-money options respectively. In the same manner, the best performance for implied trees models according to their maturity levels was in options with maturity of 30 – 90 days, followed by options maturing after 90 days, followed by options with 11 and 29 days to expiry.

5.1. Suggestions for Future Researches

Future studies might adopt a more complex function for volatility depending on strike price and time to maturity. i.e: studies can adopt quadratic function of strike and maturity. This might reduce the dispersion between models' local volatilities and market implied volatility further and increase the accuracy of both Derman and Kani and Barle and Cakici models. Another suggestion is related to the usage of different steps levels. Within this study, binomial trees models were limited to 5 steps trees. However, it is encouraged in future researches to run the tests for 10, 20 and 100 steps trees in order to get more precise calculations.

One of the problems faced during the research was the use of Microsoft Excel VBA functions. While Excel VBA is flexible and easy to-develop programming language, it is known to be very slow when executing some complex mathematical operations. The usage of Excel VBA did not considerably slow Barle and Cakici's model as it uses Black-Scholes to calculate options prices. However, the performance of Excel VBA functions was deeply affected by increasing the number of steps to 20 for Derman and Kani's model.

The final suggestion might be the use of faster development languages such as C++ to rewrite similar functions that could achieve much faster results and enable to run Derman and Kani at higher steps levels of 10, 20 and 100 steps trees.

Bibliography

- Barle, S and N. Cakici, (1998): How to Grow a Smiling Tree. The Journal of Financial Engineering, Vol. 7, No. 2, June 1998.
- Black, F. and M. Scholes (1973): The Pricing of Options and Corporate Liabilities. Journal of Political Economy, volume: 81, pp. 637 – 59.
- Chance, D. (2008). The volatility smile. Web article:
<http://www.bus.lsu.edu/academics/finance/faculty/dchance/Instructional/TN04-01.pdf>
- Cox, J., Ross, S. and Rubinstein, M. (1979). Option Pricing: A simplified Approach, Journal of Financial Economics 7: 229-263.
- Derman, E. and I. Kani (1994): The Volatility Smile and Its Implied Tree. Risk 7 no2, 32-39.
- Derman, E. (2003). Laughter in the Dark – The Problem of the Volatility Smile. Web article:
http://www.ederman.com/new/docs/euronext-volatility_smile.pdf
- Derman, E. (Lectures): Laughter in the dark – An introduction to volatility Smile. A set of unpublished lecture notes from the Master's in the Financial Engineering Program at Columbia University. Web reference: <http://www.ederman.com/new/docs/laughter.html>
- Dumas, B., Fleming, J., and Whaley, R. (1995): Implied Volatility Functions: Empirical Test. Features and Options Research Center at the Fuqua School of Business, Duke University.
- Dupire, B. (1994): Pricing with a Smile. Risk 7 no1, 18-20.
- Hardle, W., Kleinow, T., and Stahl, G. (2002): Applied Quantitative Methods: Theory and Computational Tools. Springer e-book. See www.xplore-stat.de.
- Hull, J. (2005): Fundamentals of Futures and Options Markets. 5th Edition. PEARSON, Prentice Hall.
- Hull, J. (2006): Options, Futures, and Other Derivatives. Sixth Edition. PEARSON, Prentice Hall.
- Rebel, G. (2004): Numerical Methods Project Report, Implied Binomial Trees. ICMA Center, The University of Reading. www.bgisl.com/Rebel_NMIP_Project_2004-04-19.pdf
- Rubinstein, M. (1994): Implied Binomial Trees. Journal of Finance, 69, pp. 771-818.
- Summa, J. (2004): Trading Against The Crowd, Profiting from Fear and Greed in Stock, Futures, and Options Markets. John Wiley & Sons.
- The Options Institute - Multiple Authors (1995): Options: Essential Concepts And Trading Strategies. Second Edition. The Educational Division of The Chicago Board Options Exchange. IRWIN Professional Publishing.
- Ward, R. (2004): Options and Options Trading: A Simplified Course That Takes You From Coin Tosses To Black-Scholes. McGraw Hill.
- XploRe software handbook is available on the web site: <http://fedc.wiwi.hu-berlin.de/xplore/tutorials/>

Appendix A. Dispersions For Different Moneyiness Levels

Figure A.1: ITM Call Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	18.42%	19.18%	18.98%
2	7.26%	8.53%	7.83%
3	14.04%	10.99%	9.64%
4	9.16%	4.80%	3.20%
5	15.06%	12.30%	10.86%
6	8.33%	4.78%	3.48%
7	13.65%	13.90%	13.61%
8	6.63%	2.89%	2.13%
Average	11.57%	9.67%	8.72%

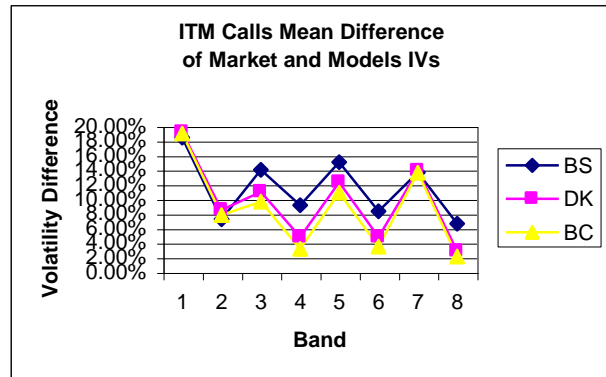


Figure A.2: NTM Call Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	8.30%	3.99%	3.30%
2	9.86%	3.57%	2.07%
3	12.78%	7.00%	5.20%
4	13.64%	7.08%	5.06%
5	12.43%	7.17%	5.80%
6	12.13%	5.70%	4.02%
7	9.55%	5.55%	4.72%
8	9.95%	3.83%	2.15%
Average	11.08%	5.49%	4.04%

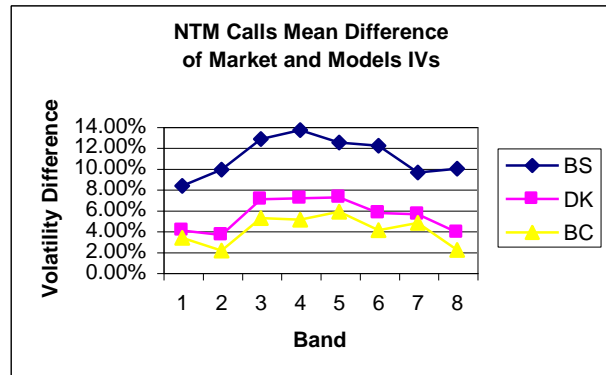


Figure A.3: OTM Call Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	12.11%	1.55%	1.43%
2	13.32%	2.34%	1.59%
3	15.76%	4.63%	3.21%
4	16.20%	5.14%	3.69%
5	15.20%	4.38%	3.06%
6	14.75%	3.82%	2.67%
7	11.76%	2.58%	2.48%
8	12.68%	1.86%	1.27%
Average	13.97%	3.29%	2.43%

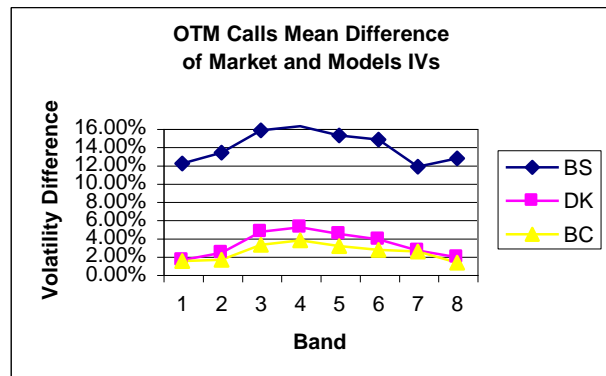


Figure A.4: ITM Put Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	12.93%	5.24%	4.95%
2	13.06%	6.24%	5.29%
3	16.27%	8.15%	6.52%
4	16.77%	6.98%	5.58%
5	15.68%	10.98%	10.02%
6	13.91%	6.13%	5.04%
7	13.98%	9.35%	8.77%
8	12.49%	3.28%	2.48%
Average	14.39%	7.04%	6.08%

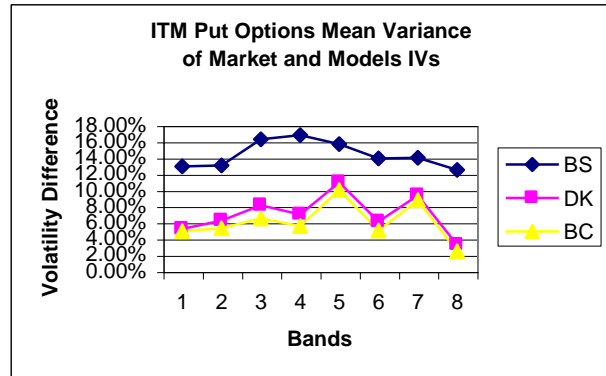


Figure A.5: NTM Put Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	9.04%	4.37%	3.45%
2	10.41%	4.05%	2.51%
3	13.17%	7.63%	5.84%
4	14.86%	8.35%	6.29%
5	13.02%	8.00%	6.58%
6	12.61%	6.29%	4.64%
7	10.35%	6.50%	5.66%
8	10.61%	4.47%	2.71%
Average	11.76%	6.21%	4.71%

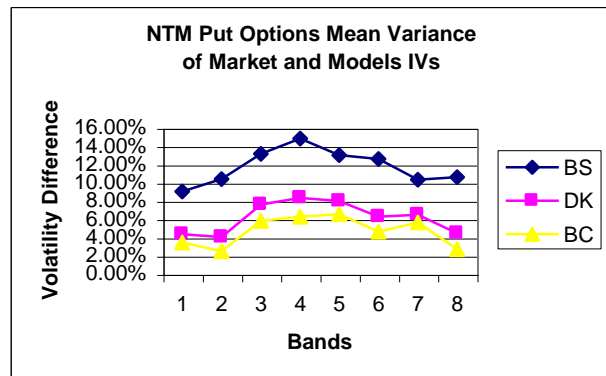
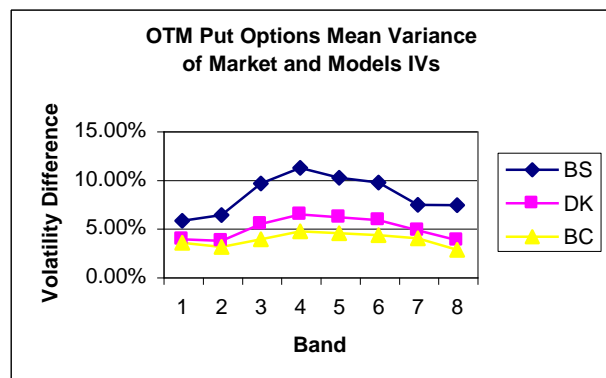


Figure A.6: OTM Put Options Mean difference of BS, DK and BC's local volatilities and market's implied volatility

Band ID	BS	DK	BC
1	5.74%	3.83%	3.49%
2	6.32%	3.68%	3.05%
3	9.56%	5.40%	3.83%
4	11.18%	6.40%	4.65%
5	10.15%	6.08%	4.46%
6	9.67%	5.79%	4.24%
7	7.35%	4.75%	3.93%
8	7.33%	3.74%	2.74%
Average	8.41%	4.96%	3.80%



Appendix B. Dispersions At Different Maturities Levels

Figure B.1: Call Options expiring after 90 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	9.99%	1.95%	1.27%
2	10.91%	2.89%	1.62%
3	13.08%	4.96%	3.20%
4	14.10%	5.70%	3.76%
5	13.98%	5.43%	3.53%
6	13.62%	5.19%	3.35%
7	10.89%	2.86%	1.60%
8	11.65%	3.14%	1.60%
Average	12.28%	4.01%	2.49%

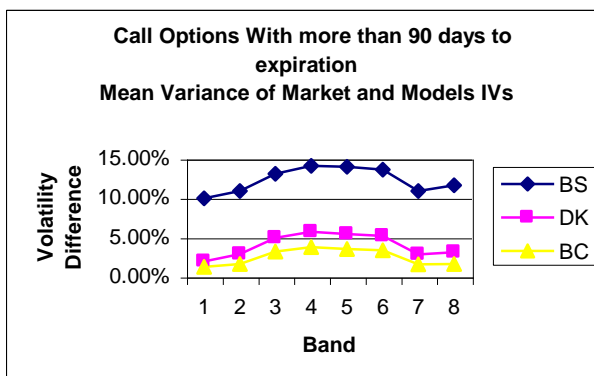


Figure B.2: Call Options expiring between 30 and 90 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	8.16%	2.13%	2.60%
2	9.54%	2.58%	2.06%
3	12.28%	4.53%	3.17%
4	14.24%	5.95%	4.28%
5	12.96%	4.42%	2.97%
6	12.80%	4.07%	2.77%
7	8.40%	2.31%	2.57%
8	9.78%	2.02%	1.60%
Average	11.02%	3.50%	2.75%

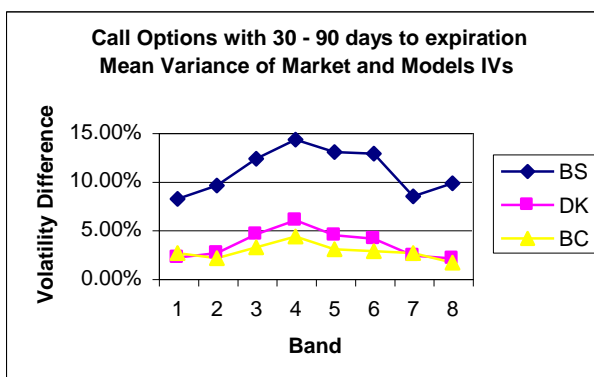


Figure B.3: Call Options expiring within less than 30 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	25.98%	27.97%	27.84%
2	10.29%	15.10%	14.24%
3	32.91%	34.15%	33.81%
4	11.06%	4.61%	3.53%
5	18.81%	19.54%	19.79%
6	8.19%	4.19%	4.02%
7	26.65%	39.26%	40.68%
8	8.03%	2.55%	2.53%
Average	17.74%	18.42%	18.31%

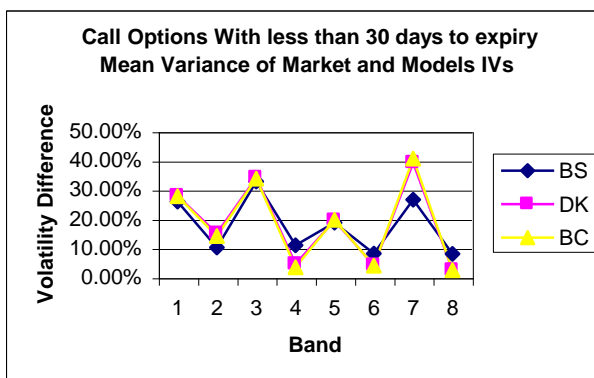


Figure B.4: Put Options expiring after 90 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	10.05%	3.31%	1.96%
2	11.17%	4.17%	2.52%
3	13.19%	6.71%	4.76%
4	15.22%	8.38%	6.41%
5	14.67%	7.75%	5.73%
6	14.03%	6.95%	5.00%
7	11.33%	4.39%	2.76%
8	11.54%	4.69%	2.91%
Average	12.65%	5.79%	4.00%

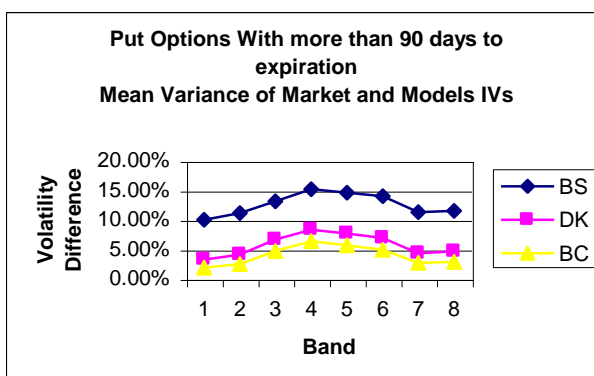


Figure B.5: Put Options expiring between 30 and 90 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	6.86%	2.54%	3.03%
2	8.20%	2.58%	1.97%
3	11.11%	4.55%	3.07%
4	13.75%	6.42%	4.64%
5	12.26%	4.78%	3.28%
6	11.68%	4.24%	2.93%
7	7.52%	2.52%	2.78%
8	8.46%	2.57%	2.02%
Average	9.98%	3.77%	2.96%

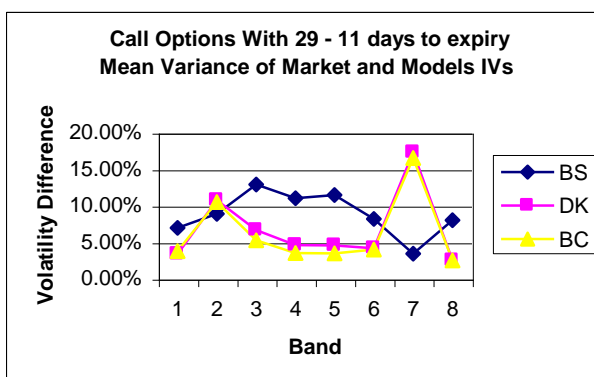


Figure B.6: Put Options expiring within less than 30 days - Mean difference of BS, DK and BC's local volatilities and market's IV

Band ID	BS	DK	BC
1	8.40%	8.85%	9.38%
2	8.00%	9.35%	9.28%
3	13.15%	17.07%	16.65%
4	12.27%	6.23%	5.10%
5	10.58%	17.26%	17.73%
6	7.78%	7.28%	7.01%
7	16.37%	31.84%	31.22%
8	7.78%	4.01%	3.60%
Average	10.54%	12.74%	12.50%

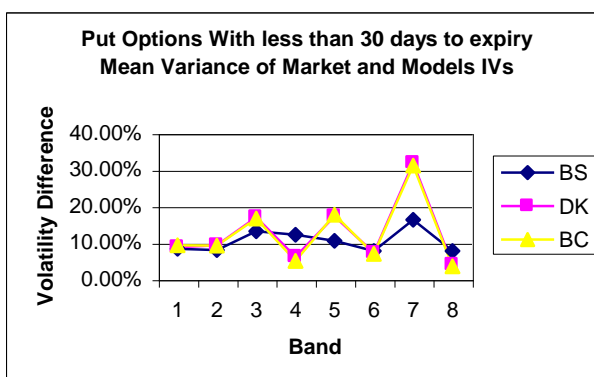


Figure B.7: Call Options expiring within 11 to 29 days - Mean difference of BS, DK and BC's volatilities and market's IV

Band ID	BS	DK	BC
1	6.98%	3.42%	3.80%
2	8.89%	10.77%	10.51%
3	12.94%	6.67%	5.29%
4	11.06%	4.61%	3.53%
5	11.47%	4.58%	3.47%
6	8.19%	4.19%	4.02%
7	3.46%	17.35%	16.56%
8	8.03%	2.55%	2.53%
Average	8.88%	6.77%	6.21%

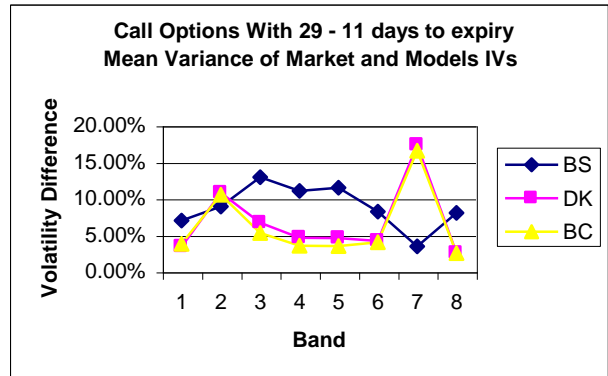
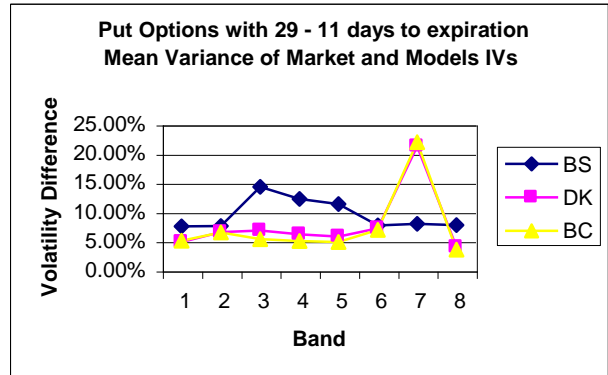


Figure B.8: Put Options expiring within 11 – 29 days - Mean difference of BS, DK and BC's volatilities and market's IV

Band ID	BS	DK	BC
1	7.59%	4.90%	5.13%
2	7.63%	6.60%	6.54%
3	14.37%	6.89%	5.39%
4	12.27%	6.23%	5.10%
5	11.40%	5.84%	4.90%
6	7.78%	7.28%	7.01%
7	8.04%	21.27%	22.01%
8	7.78%	4.01%	3.60%
Average	9.61%	7.88%	7.46%



Appendix C. Excel VBA Functions Specifications

Table C.1: Excel VBA Functions Specifications

Function ID	Function Name	Inputs	Output
1	BC	Underlier: asset price Strik: strike price Maturity: maturity of option Risk_Free: risk free rate Dividend: dividend paid Steps: number of steps Option_Type: 0 for call and 1 for put	Option Price according to Barle and Cakici method.
2	BS_Call	Underlier: asset price Strik: strike price Maturity: maturity of option BS_Volatility: the constant volatility Risk_Free: risk free rate Dividend: dividend paid	Call Option Price according to Black and Scholes model.
3	BS_IV	Underlier: asset price Strik: strike price Maturity: maturity of option Risk_Free: risk free rate Dividend: dividend paid Option_Price: market price of the option Tol: the precision required. i.e.: 0.0001 Call_or_Put: 0 for call and 1 for put	The implied volatility resulting from observed option price, or option price calculated by DK or BC models.
4	BS_Put	Underlier: asset price Strik: strike price Maturity: maturity of option BS_Volatility: the constant volatility Risk_Free: risk free rate Dividend: dividend paid	Put Option Price according to Black and Scholes model.
5	CRR	Underlier: asset price Strik: strike price Maturity: maturity of option BS_Volatility: the constant volatility Risk_Free: risk free rate Dividend: dividend paid Steps: number of steps Option_Type: 0 for call and 1 for put	Option Price according to Cox-Ross-Rubinstein binomial tree method.
6	DK	Underlier: asset price Strik: strike price Maturity: maturity of option Risk_Free: risk free rate Dividend: dividend paid Steps: number of steps Option_Type: 0 for call and 1 for put	Option Price according to Derman and Kani method.

Function ID	Function Name	Inputs	Output
7	Vol_Smile	Underlier: asset price Strik: strike price	Returns options volatility according to the volatility function defined within the code

Appendix D. Excel VBA Functions Code

Function D.1: BC

Function BC(Underlier, Strik, Maturity, Risk_Free, Dividend, Steps, Option_Type)

' Variables definition

Dim TimeStep, time_to_ex, inf_no_dividend, inf_with_dividend, T_node, K_temp, S_temp

Dim Opt_Price, Out_Opt_Price, rho_sum, S_numer, S_denom

Dim i As Integer, i_sta As Integer, i_bel As Integer, i_abo As Integer, j As Integer, i_sum As Integer

Dim arr_Output_Data, Cum_sum

Dim row_count As Integer

Dim V As Integer, w As Integer

Dim mean_log_price_sum

Dim log_var_sum

Dim Call_Sum, Put_Sum

' -----

' Matrix definition

' -----

ReDim Forward_Price(0 To Steps, 0 To Steps) ' Forward prices at each node

ReDim Local_impl_vol_TimeStep(0 To Steps - 1, 0 To Steps - 1) ' note the reduced size of this array

Dim max_impl_vol_rows As Integer

' We need to limit the size of the Local_impl_vol matrix, note that max_impl_vol_rows becomes very large

If Steps <= 20 Then

max_impl_vol_rows = Steps * (Steps + 1) * (Steps + 2) / 6 - 1 ' i.e. from loop For j = 0 To Steps - 1, For i = 0 To j, For w = j + 1 To n

```

    ReDim Local_impl_vol(0 To max_impl_vol_rows, 0 To 2)
    ReDim arr_Node_Probabilities_temp(0 To Steps, 0 To Steps) ' Temp probabilities of reaching each node, used in the calculation of
Local_impl_vol(...)
End If

ReDim arr_Node_State_Prices(0 To Steps, 0 To Steps) ' Arrow-Debreu state prices
ReDim arr_Node_Probabilities(0 To Steps, 0 To Steps) ' Probabilities of reaching each node in the tree
ReDim arr_Output_Data(0 To Steps, 0 To Steps) ' temp array for formatting output data
ReDim arr_Node_Probabilities_trans(0 To Steps - 1, 0 To Steps - 1) ' Transition probabilities at each node, note the reduced size of this array
ReDim arr_Stock_Prices(0 To Steps, 0 To Steps) ' Stock prices at each node
' -----
' Tree parameters (note the continuous compounding)
' -----
TimeStep = Maturity / Steps
inf_no_dividend = Exp(Risk_Free * TimeStep)
inf_with_dividend = Exp((Risk_Free - Dividend) * TimeStep)
' -----
' Set the starting values at the base of the tree
' -----
arr_Stock_Prices(0, 0) = Underlier
arr_Node_State_Prices(0, 0) = 1
arr_Node_Probabilities(0, 0) = 1
T_node = 0

```

```

' -----
' Calculate DK and BC tree node stock prices and other values
' -----
For j = 1 To Steps ' the time steps

    T_node = T_node + TimeStep ' time at each node
    ' -----
    ' Calculate the forward prices at the vertical nodes
    ' -----
    For i = 0 To j - 1 ' vertical nodes

        Forward_Price(i, j) = arr_Stock_Prices(i, j - 1) * inf_with_dividend

    Next i

    ' -----
    ' Determine the vertical nodes for calculating the central, above and below tree data
    ' -----
    i_sta = Int((j + 0.0001) / 2) ' central nodes
    i_bel = Int((j + 1.0001) / 2 + 1) ' nodes below the centre (decreasing stock prices)

    If i_bel > j Then i_bel = j

```

i_abo = i_sta - 1 ' nodes above the centre (increasing stock prices)

If i_abo < 0 Then i_abo = 0

' -----
' Calculate the tree parameters for the nodes at the centre of the tree
' -----

i = i_sta

If j Mod 2 = 0 Then ' i.e. j is even so we have only one central node

arr_Stock_Prices(i, j) = Underlier * (inf_with_dividend ^ j)

' -----
' Compensation against arbitrage and ensure $0 < \text{arr_Node_Probabilities_trans} < 1$
' Applied Quantitative Finance PDF Book, Equation (7.15)
' -----

If i > 0 And i < j Then

If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2 ' i.e. the average of the two

End If

End If

Else ' j is odd and there are two central nodes

K_temp = Forward_Price(i, j)

S_temp = K_temp

Opt_Price = BS_Call(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend)

rho_sum = 0

If j > 2 Then

For i_sum = 0 To i - 1 ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)

rho_sum = rho_sum + arr_Node_State_Prices(i_sum, j - 1) * (Forward_Price(i_sum, j) - K_temp)

Next i_sum

End If

S_numer = S_temp * (inf_no_dividend * Opt_Price + arr_Node_State_Prices(i, j - 1) * K_temp - rho_sum)

S_denom = arr_Node_State_Prices(i, j - 1) * Forward_Price(i, j) - inf_no_dividend * Opt_Price + rho_sum

arr_Stock_Prices(i, j) = S_numer / S_denom ' Applied Quantitative Finance PDF Book, Equation (7.11 and 7.18)

arr_Stock_Prices(i + 1, j) = S_temp * S_temp / arr_Stock_Prices(i, j)

' -----

' Compensation against arbitrage and ensure $0 < \text{arr_Node_Probabilities_trans} < 1$

' Applied Quantitative Finance PDF Book, Equation (7.15)

' -----

If i > 0 And i < j Then

If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2

```

End If
If arr_Stock_Prices(i + 1, j) <= Forward_Price(i + 1, j) Or arr_Stock_Prices(i + 1, j) >= Forward_Price(i, j) Then
    arr_Stock_Prices(i + 1, j) = (Forward_Price(i + 1, j) + Forward_Price(i, j)) / 2
End If
End If
End If

' -----
' Calculate the tree parameters for the nodes above the centre (increasing stock prices)
' -----

For i = i_abo To 0 Step -1 ' vertical nodes

    K_temp = Forward_Price(i, j)
    Opt_Price = BS_Call(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend)

    S_temp = arr_Stock_Prices(i + 1, j)
    rho_sum = 0
    For i_sum = 0 To i - 1 ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)
        rho_sum = rho_sum + arr_Node_State_Prices(i_sum, j - 1) * (Forward_Price(i_sum, j) - K_temp)
    Next i_sum

    S_numer = S_temp * (inf_no_dividend * Opt_Price - rho_sum) - arr_Node_State_Prices(i, j - 1) * K_temp * (Forward_Price(i, j) - S_temp)
    S_denom = inf_no_dividend * Opt_Price - rho_sum - arr_Node_State_Prices(i, j - 1) * (Forward_Price(i, j) - S_temp)

```

```

arr_Stock_Prices(i, j) = S_numer / S_denom ' Applied Quantitative Finance PDF Book, Equation (7.13 and 7.20)
' -----
' Compensation against arbitrage and ensure 0 < arr_Node_Probabilities_trans < 1
' Applied Quantitative Finance PDF Book, Equation (7.15)
' -----

If j > 1 And i = 0 Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Then
    arr_Stock_Prices(i, j) = arr_Stock_Prices(i + 1, j) * Forward_Price(i, j) / Forward_Price(i + 1, j)
  End If
End If

If i > 0 And i < j Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

    arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2
  End If
End If

Next i
' -----
' Calculate the tree parameters for the nodes below the centre (decreasing stock prices)
' -----

For i = i_bel To j ' vertical nodes

  K_temp = Forward_Price(i - 1, j)

```

Opt_Price = BS_Put(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend) ' Note Put Option

S_temp = arr_Stock_Prices(i - 1, j)

rho_sum = 0

For i_sum = i To j - 1 ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)

 rho_sum = rho_sum + arr_Node_State_Prices(i_sum, j - 1) * (K_temp - Forward_Price(i_sum, j))

Next i_sum

S_numer = S_temp * (inf_no_dividend * Opt_Price - rho_sum) + arr_Node_State_Prices(i - 1, j - 1) * K_temp * (Forward_Price(i - 1, j) - S_temp)

S_denom = inf_no_dividend * Opt_Price - rho_sum + arr_Node_State_Prices(i - 1, j - 1) * (Forward_Price(i - 1, j) - S_temp)

arr_Stock_Prices(i, j) = S_numer / S_denom ' Applied Quantitative Finance PDF Book, Equation (7.14 and 7.21)

' -----

' Compensation against arbitrage and ensure $0 < \text{arr_Node_Probabilities_trans} < 1$

' Applied Quantitative Finance PDF Book, Equation (7.15)

' -----

If i > 0 And i < j Then

 If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

 arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2

 End If

End If

If i > 1 And i = j Then

 If arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

 arr_Stock_Prices(i, j) = arr_Stock_Prices(i - 1, j) * Forward_Price(i - 1, j) / Forward_Price(i - 2, j)

```

    End If
  End If
Next i
' -----
' Calculate the transition probabilities
' Applied Quantitative Finance PDF Book, Equation (7.5)
' Calculate the implied local volatilities the time to expiry = the time step, TimeStep = T/n
' -----
For i = 0 To j - 1 ' vertical nodes
  arr_Node_Probabilities_trans(i, j - 1) = (Forward_Price(i, j) - arr_Stock_Prices(i + 1, j)) / (arr_Stock_Prices(i, j) - arr_Stock_Prices(i + 1, j))
  If arr_Node_Probabilities_trans(i, j - 1) <= 0 Or arr_Node_Probabilities_trans(i, j - 1) >= 1 Then
    MsgBox "Caution: arr_Node_Probabilities_trans(" & i & ", " & j - 1 & ")=" & arr_Node_Probabilities_trans(i, j - 1)
  End If
  ' This is for the time to expiry = the time step TimeStep, see page 152 Applied Quantitative Finance PDF Book, just below equation (7.16):
  Local_impl_vol_TimeStep(i, j - 1) = (arr_Node_Probabilities_trans(i, j - 1) * (1 - arr_Node_Probabilities_trans(i, j - 1))) ^ 0.5 *
Log(arr_Stock_Prices(i, j) / arr_Stock_Prices(i + 1, j))
Next i
' -----
' Calculate the Arrow-Debreu state prices Applied Quantitative Finance PDF Book, Equations (7.6)
' -----
arr_Node_State_Prices(j, j) = arr_Node_State_Prices(j - 1, j - 1) * (1 - arr_Node_Probabilities_trans(j - 1, j - 1)) / inf_no_dividend
For i = j - 1 To 1 Step -1 ' vertical nodes
  arr_Node_State_Prices(i, j) = (arr_Node_State_Prices(i, j - 1) * _

```

```

arr_Node_Probabilities_trans(i, j - 1) + arr_Node_State_Prices(i - 1, j - 1) * (1 - arr_Node_Probabilities_trans(i - 1, j - 1)) / inf_no_dividend
Next i
arr_Node_State_Prices(0, j) = arr_Node_State_Prices(0, j - 1) * arr_Node_Probabilities_trans(0, j - 1) / inf_no_dividend
' -----
' Calculate the probabilities of reaching each node in the tree
' -----

For i = 0 To j ' vertical nodes
arr_Node_Probabilities(i, j) = arr_Node_State_Prices(i, j) * inf_no_dividend ^ j ' from the definition of Arrow-Debreu state price
Next i
Next j
' -----
' Calculate the local volatilities implied by the implied binomial tree
' See : http://www.xplora-stat.de/help/IBTlocsigma.html for comparison
' See also page 152 Applied Quantitative Finance PDF Book, Equation (7.17)
' This implementation assumes that the temp node probability arr_Node_Probabilities_temp(i, j) = 1
' where (i,j) corresponds to the node (i.e. stock price) for which the implied
' volatility is being calculated i.e. arr_Stock_Prices(i, j)
' -----
If Steps <= 20 Then ' otherwise the size of the Local_impl_vol array becomes too large, see max_impl_vol_rows
row_count = 0
For j = 0 To Steps - 1 ' time steps
For i = 0 To j ' vertical nodes

```

```

time_to_ex = TimeStep ' used later in the output array
ReDim arr_Node_Probabilities_temp(0 To Steps, 0 To Steps) ' this is required to clear the contenTimeStep of arr_Node_Probabilities_temp,
do not move it to top of function
arr_Node_Probabilities_temp(i, j) = 1 ' we assume that the probability of being at arr_Stock_Prices(i, j) is 1
For w = j + 1 To Steps ' the possible time steps (or poinTimeStep) at which local volatilities can be calculated
    ' Calculate the temporary node probabilities
    arr_Node_Probabilities_temp(i, w) = arr_Node_Probabilities_temp(i, w - 1) * arr_Node_Probabilities_trans(i, w - 1)
    For V = i + 1 To w - 1
        arr_Node_Probabilities_temp(V, w) = arr_Node_Probabilities_temp(V, w - 1) * arr_Node_Probabilities_trans(V, w - 1) + _
            arr_Node_Probabilities_temp(V - 1, w - 1) * (1 - arr_Node_Probabilities_trans(V - 1, w - 1))
    Next V
    arr_Node_Probabilities_temp(w, w) = arr_Node_Probabilities_temp(w - 1, w - 1) * (1 - arr_Node_Probabilities_trans(w - 1, w - 1))
    ' Calculate the expected value of the log stock price, Applied Quantitative Finance PDF Book, Equation (7.17)
    mean_log_price_sum = 0
    For V = i To w
        mean_log_price_sum = mean_log_price_sum + arr_Node_Probabilities_temp(V, w) * Log(arr_Stock_Prices(V, w))
    Next V

    ' Calculate the log variance, Applied Quantitative Finance PDF Book, Equation (7.17)
    log_var_sum = 0
    For V = i To w
        log_var_sum = log_var_sum + arr_Node_Probabilities_temp(V, w) * (Log(arr_Stock_Prices(V, w)) - mean_log_price_sum) ^ 2
    Next V

```

```

' Write the calculated values into the output array
Local_impl_vol(row_count, 0) = arr_Stock_Prices(i, j) ' stock price
Local_impl_vol(row_count, 1) = time_to_ex ' time to expiry for calculation of the implied local volatility
Local_impl_vol(row_count, 2) = log_var_sum ^ 0.5 ' implied local volatility from variance
row_count = row_count + 1
time_to_ex = time_to_ex + TimeStep
Next w
Next i
Next j
End If ' Steps <= 20
' -----
' Calculate the price of the call using the terminal AD state
' prices and the terminal call payoffs
' Applied Quantitative Finance PDF Book, Equations (7.9) and (7.10)
' -----
Call_Sum = 0
Put_Sum = 0

For i = 0 To Steps ' terminal vertical nodes
    Call_Sum = Call_Sum + arr_Node_State_Prices(i, Steps) * Application.Max(arr_Stock_Prices(i, Steps) - Strik, 0)
    Put_Sum = Put_Sum + arr_Node_State_Prices(i, Steps) * Application.Max(Strik - arr_Stock_Prices(i, Steps), 0)
Next i

```


If Option_Type = 0 Then ' Call option

 Out_Opt_Price = Call_Sum

Else ' Put option

 Out_Opt_Price = Put_Sum

End If

BC = Out_Opt_Price

End Function

Function D.2: BS_Call

Function BS_Call(Underlier, Strik, Maturity, BS_Volatility, Risk_Free, Dividend)

 Dim d1, d2

 Dim Nd1, Nd2

 d1 = (Log(Underlier / Strik) + (Risk_Free - Dividend + 0.5 * BS_Volatility ^ 2) * Maturity) / (BS_Volatility * Maturity ^ 0.5)

 d2 = d1 - (BS_Volatility * Maturity ^ 0.5)

 Nd1 = Application.NormSDist(d1)

 Nd2 = Application.NormSDist(d2)

 BS_Call = Underlier * Exp(-Dividend * Maturity) * Nd1 - Strik * Exp(-Risk_Free * Maturity) * Nd2

End Function

Function D.3: BS_IV

```
Function BS_IV(Underlier, Strik, Maturity, Risk_Free, Dividend, Option_Price, Tol, Call_or_Put)
  Dim High_sigma, Low_sigma
  High_sigma = 2 ' i.e. 200%
  Low_sigma = 0
  If Call_or_Put = 0 Then ' Call Option
    Do While (High_sigma - Low_sigma) > Tol
      If BS_Call(Underlier, Strik, Maturity, (High_sigma + Low_sigma) / 2, Risk_Free, Dividend) > Option_Price Then
        High_sigma = (High_sigma + Low_sigma) / 2
      Else
        Low_sigma = (High_sigma + Low_sigma) / 2
      End If
    Loop

  Else ' Put Option
    Do While (High_sigma - Low_sigma) > Tol
      If BS_Put(Underlier, Strik, Maturity, (High_sigma + Low_sigma) / 2, Risk_Free, Dividend) > Option_Price Then
        High_sigma = (High_sigma + Low_sigma) / 2
      Else
        Low_sigma = (High_sigma + Low_sigma) / 2
      End If
    Loop
  End If
  BS_IV = (High_sigma + Low_sigma) / 2
End Function
```

Function D.4: BS_Put

```
Function BS_Put(Underlier, Strik, Maturity, BS_Volatility, Risk_Free, Dividend)
  Dim d1, d2
  Dim Nd1, Nd2
  d1 = (Log(Underlier / Strik) + (Risk_Free - Dividend + 0.5 * BS_Volatility ^ 2) * Maturity) / (BS_Volatility * Maturity ^ 0.5)
  d2 = d1 - (BS_Volatility * Maturity ^ 0.5)
  Nd1 = Application.NormSDist(-d1)
  Nd2 = Application.NormSDist(-d2)
  BS_Put = Strik * Exp(-Risk_Free * Maturity) * Nd2 - Underlier * Exp(-Dividend * Maturity) * Nd1
End Function
```

Function D.5: CRR

```
Function CRR(Underlier, Strik, Maturity, BS_Volatility, Risk_Free, Dividend, Steps, Option_Type)
  ' variables definition
  Dim TimeStep ' Time Step
  Dim i As Integer 'counter
  Dim j As Integer 'counter
  Dim Up, Down ' Move up and down factors
  Dim q ' risk neutral probability
  Dim Discount ' discount factor
  Dim Call_Sum, Put_Sum
  Dim Opt_Price ' option price

  ' Arrays definition
  ReDim arr_Stock_Prices(0 To Steps, 0 To Steps) ' Stock Prices Matrix
  ReDim arr_Node_Probabilities(0 To Steps, 0 To Steps) ' Node Probabilities Matrix
  ReDim arr_Node_State_Prices(0 To Steps, 0 To Steps) ' Node State Prices Matrix
  ReDim arr_Terminal_Values(0 To Steps) ' Terminal Call Values Matrix

  TimeStep = Maturity / Steps
  Up = Exp(BS_Volatility * TimeStep ^ 0.5) ' up movement
  Down = 1 / Up ' down movement
```

q = (Exp((Risk_Free - Dividend) * TimeStep) - Down) / (Up - Down) ' transition probability at every node
Discount = Exp(-Risk_Free * TimeStep) ' discount factor

```

Call_Sum = 0
For j = 0 To Steps ' time steps to maturity
  For i = 0 To j ' vertical nodes
    arr_Stock_Prices(i, j) = Underlier * (Down ^ i) * Up ^ (j - i) ' node stock prices
    arr_Node_Probabilities(i, j) = Application.Combin(j, i) * ((1 - q) ^ i) * q ^ (j - i) ' node probabilities
    arr_Node_State_Prices(i, j) = arr_Node_Probabilities(i, j) * Discount ^ j ' node AD prices or state prices
    If j = Steps Then ' determine the terminal call payoffs
      arr_Terminal_Values(i) = Application.Max(arr_Stock_Prices(i, j) - Strik, 0) ' terminal call payoffs
      Call_Sum = Call_Sum + arr_Terminal_Values(i) * arr_Node_State_Prices(i, j) ' sum of product of terminal payoffs and terminal state prices
    End If
  Next i
Next j

```

```

If Option_Type = 0 Then ' Call Option
  Opt_Price = Call_Sum
Else ' Put Option
  Opt_Price = Call_Sum + Strik * Exp(-Risk_Free * Maturity) - Underlier * Exp(-Dividend * Maturity) ' from the put call parity relationship
End If

```

```

  CRR = Opt_Price
End Function

```

Function D.6: DK

Function DK(Underlier, Strik, Maturity, Risk_Free, Dividend, Steps, Option_Type)

' Variables definition

```

Dim TimeStep, time_to_ex, inf_no_dividend, inf_with_dividend, T_node, K_temp, S_temp
Dim Opt_Price, Out_Opt_Price, rho_sum, S_numer, S_denom
Dim i As Integer, i_sta As Integer, i_bel As Integer, i_abo As Integer, j As Integer, i_sum As Integer
Dim arr_Output_Data, Cum_sum
Dim row_count As Integer

```

```

Dim V As Integer, w As Integer
Dim mean_log_price_sum
Dim log_var_sum
Dim Call_Sum, Put_Sum
' -----
' Matrix definition
' -----
ReDim Forward_Price(0 To Steps, 0 To Steps) ' Forward prices at each node
ReDim Local_impl_vol_TimeStep(0 To Steps - 1, 0 To Steps - 1) ' note the reduced size of this array
Dim max_impl_vol_rows As Integer
' We need to limit the size of the Local_impl_vol matrix, note that max_impl_vol_rows becomes very large
If Steps <= 20 Then

    max_impl_vol_rows = Steps * (Steps + 1) * (Steps + 2) / 6 - 1 ' i.e. from loop For j = 0 To Steps - 1, For i = 0 To j, For w = j + 1 To n
    ReDim Local_impl_vol(0 To max_impl_vol_rows, 0 To 2)
    ReDim arr_Node_Probabilities_temp(0 To Steps, 0 To Steps) ' Temp probabilities of reaching each node, used in the calculation of
Local_impl_vol(...)
End If

ReDim arr_Node_State_Prices(0 To Steps, 0 To Steps) ' Arrow-Debreu state prices
ReDim arr_Node_Probabilities(0 To Steps, 0 To Steps) ' Probabilities of reaching each node in the tree
ReDim arr_Output_Data(0 To Steps, 0 To Steps) ' temp array for formatting output data
ReDim arr_Node_Probabilities_trans(0 To Steps - 1, 0 To Steps - 1) ' Transition probabilities at each node, note the reduced size of this array
ReDim arr_Stock_Prices(0 To Steps, 0 To Steps) ' Stock prices at each node
' -----
' Tree parameters (note the continuous compounding)
' -----
TimeStep = Maturity / Steps
inf_no_dividend = Exp(Risk_Free * TimeStep)
inf_with_dividend = Exp((Risk_Free - Dividend) * TimeStep)
' -----
' Set the starting values at the base of the tree
' -----
arr_Stock_Prices(0, 0) = Underlier

```

```

arr_Node_State_Prices(0, 0) = 1
arr_Node_Probabilities(0, 0) = 1
T_node = 0
' -----
' Calculate DK and BC tree node stock prices and other values
' -----
For j = 1 To Steps ' the time steps

    T_node = T_node + TimeStep ' time at each node
    ' -----
    ' Calculate the forward prices at the vertical nodes
    ' -----
    For i = 0 To j - 1 ' vertical nodes

        Forward_Price(i, j) = arr_Stock_Prices(i, j - 1) * inf_with_dividend

    Next i

    ' -----
    ' Determine the vertical nodes for calculating the central, above and below tree data
    ' -----
    i_sta = Int((j + 0.0001) / 2) ' central nodes
    i_bel = Int((j + 1.0001) / 2 + 1) ' nodes below the centre (decreasing stock prices)

    If i_bel > j Then i_bel = j

    i_abo = i_sta - 1 ' nodes above the centre (increasing stock prices)

    If i_abo < 0 Then i_abo = 0

    ' -----
    ' Calculate the tree parameters for the nodes at the centre of the tree
    ' -----
    i = i_sta

```

If $j \bmod 2 = 0$ Then ' i.e. j is even so we have only one central node

```
arr_Stock_Prices(i, j) = Underlier
```

```
' -----  
' Compensation against arbitrage and ensure  $0 < arr\_Node\_Probabilities\_trans < 1$   
' Applied Quantitative Finance PDF Book, Equation (7.15)  
' -----
```

```
If  $i > 0$  And  $i < j$  Then
```

```
  If  $arr\_Stock\_Prices(i, j) \leq Forward\_Price(i, j)$  Or  $arr\_Stock\_Prices(i, j) \geq Forward\_Price(i - 1, j)$  Then
```

```
     $arr\_Stock\_Prices(i, j) = (Forward\_Price(i, j) + Forward\_Price(i - 1, j)) / 2$  ' i.e. the average of the two
```

```
  End If
```

```
End If
```

Else ' j is odd and there are two central nodes

```
K_temp = arr_Stock_Prices(i, j - 1)
```

```
S_temp = Underlier
```

```
Opt_Price = CRR(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend, j, 0)
```

```
rho_sum = 0
```

```
If  $j > 2$  Then
```

```
  For  $i\_sum = 0$  To  $i - 1$  ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)
```

```
     $rho\_sum = rho\_sum + arr\_Node\_State\_Prices(i\_sum, j - 1) * (Forward\_Price(i\_sum, j) - K\_temp)$ 
```

```
  Next  $i\_sum$ 
```

```
End If
```

```
S_number = S_temp * (inf_no_dividend * Opt_Price + arr_Node_State_Prices(i, j - 1) * K_temp - rho_sum)
```

```
S_denom = arr_Node_State_Prices(i, j - 1) * Forward_Price(i, j) - inf_no_dividend * Opt_Price + rho_sum
```

```
 $arr\_Stock\_Prices(i, j) = S\_number / S\_denom$  ' Applied Quantitative Finance PDF Book, Equation (7.11 and 7.18)
```

```
 $arr\_Stock\_Prices(i + 1, j) = S\_temp * S\_temp / arr\_Stock\_Prices(i, j)$ 
```

```
' -----  
' Compensation against arbitrage and ensure  $0 < arr\_Node\_Probabilities\_trans < 1$   
' Applied Quantitative Finance PDF Book, Equation (7.15)
```

```

' -----
If i > 0 And i < j Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then
    arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2
  End If
  If arr_Stock_Prices(i + 1, j) <= Forward_Price(i + 1, j) Or arr_Stock_Prices(i + 1, j) >= Forward_Price(i, j) Then
    arr_Stock_Prices(i + 1, j) = (Forward_Price(i + 1, j) + Forward_Price(i, j)) / 2
  End If
End If
End If

' -----
' Calculate the tree parameters for the nodes above the centre (increasing stock prices)
' -----
For i = i_abo To 0 Step -1 ' vertical nodes

  K_temp = arr_Stock_Prices(i, j - 1)
  Opt_Price = CRR(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend, j, 0)

  S_temp = arr_Stock_Prices(i + 1, j)
  rho_sum = 0
  For i_sum = 0 To i - 1 ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)
    rho_sum = rho_sum + arr_Node_State_Prices(i_sum, j - 1) * (Forward_Price(i_sum, j) - K_temp)
  Next i_sum

  S_numer = S_temp * (inf_no_dividend * Opt_Price - rho_sum) - arr_Node_State_Prices(i, j - 1) * K_temp * (Forward_Price(i, j) - S_temp)
  S_denom = inf_no_dividend * Opt_Price - rho_sum - arr_Node_State_Prices(i, j - 1) * (Forward_Price(i, j) - S_temp)
  arr_Stock_Prices(i, j) = S_numer / S_denom ' Applied Quantitative Finance PDF Book, Equation (7.13 and 7.20)
' -----
' Compensation against arbitrage and ensure 0 < arr_Node_Probabilities_trans < 1
' Applied Quantitative Finance PDF Book, Equation (7.15)
' -----
If j > 1 And i = 0 Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Then

```



```

arr_Stock_Prices(i, j) = arr_Stock_Prices(i + 1, j) * Forward_Price(i, j) / Forward_Price(i + 1, j)
End If
End If
If i > 0 And i < j Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then

    arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2
  End If
End If
Next i
'-----
' Calculate the tree parameters for the nodes below the centre (decreasing stock prices)
'-----
For i = i_bel To j ' vertical nodes

  K_temp = arr_Stock_Prices(i - 1, j - 1)
  Opt_Price = CRR(Underlier, K_temp, T_node, Vol_Smile(Underlier, K_temp), Risk_Free, Dividend, j, 1) ' Note Put Option

  S_temp = arr_Stock_Prices(i - 1, j)
  rho_sum = 0
  For i_sum = i To j - 1 ' Applied Quantitative Finance PDF Book, Equation (7.12 and 7.19)
    rho_sum = rho_sum + arr_Node_State_Prices(i_sum, j - 1) * (K_temp - Forward_Price(i_sum, j))
  Next i_sum
  S_numer = S_temp * (inf_no_dividend * Opt_Price - rho_sum) + arr_Node_State_Prices(i - 1, j - 1) * K_temp * (Forward_Price(i - 1, j) - S_temp)
  S_denom = inf_no_dividend * Opt_Price - rho_sum + arr_Node_State_Prices(i - 1, j - 1) * (Forward_Price(i - 1, j) - S_temp)
  arr_Stock_Prices(i, j) = S_numer / S_denom ' Applied Quantitative Finance PDF Book, Equation (7.14 and 7.21)
  '-----
  ' Compensation against arbitrage and ensure 0 < arr_Node_Probabilities_trans < 1
  ' Applied Quantitative Finance PDF Book, Equation (7.15)
  '-----
If i > 0 And i < j Then
  If arr_Stock_Prices(i, j) <= Forward_Price(i, j) Or arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then
    arr_Stock_Prices(i, j) = (Forward_Price(i, j) + Forward_Price(i - 1, j)) / 2
  End If

```

```

End If

If i > 1 And i = j Then
    If arr_Stock_Prices(i, j) >= Forward_Price(i - 1, j) Then
        arr_Stock_Prices(i, j) = arr_Stock_Prices(i - 1, j) * Forward_Price(i - 1, j) / Forward_Price(i - 2, j)
    End If
End If

Next i
'-----
' Calculate the transition probabilities
' Applied Quantitative Finance PDF Book, Equation (7.5)
' Calculate the implied local volatilities the time to expiry = the time step, TimeStep = T/n
'-----
For i = 0 To j - 1 ' vertical nodes
    arr_Node_Probabilities_trans(i, j - 1) = (Forward_Price(i, j) - arr_Stock_Prices(i + 1, j)) / (arr_Stock_Prices(i, j) - arr_Stock_Prices(i + 1, j))
    If arr_Node_Probabilities_trans(i, j - 1) <= 0 Or arr_Node_Probabilities_trans(i, j - 1) >= 1 Then
        MsgBox "Caution: arr_Node_Probabilities_trans(" & i & ", " & j - 1 & ")= " & arr_Node_Probabilities_trans(i, j - 1)
    End If
    ' This is for the time to expiry = the time step TimeStep, see page 152 Applied Quantitative Finance PDF Book, just below equation (7.16):
    Local_impl_vol_TimeStep(i, j - 1) = (arr_Node_Probabilities_trans(i, j - 1) * (1 - arr_Node_Probabilities_trans(i, j - 1))) ^ 0.5 *
    Log(arr_Stock_Prices(i, j) / arr_Stock_Prices(i + 1, j))
Next i
'-----
' Calculate the Arrow-Debreu state prices Applied Quantitative Finance PDF Book, Equations (7.6)
'-----
arr_Node_State_Prices(j, j) = arr_Node_State_Prices(j - 1, j - 1) * (1 - arr_Node_Probabilities_trans(j - 1, j - 1)) / inf_no_dividend
For i = j - 1 To 1 Step -1 ' vertical nodes
    arr_Node_State_Prices(i, j) = (arr_Node_State_Prices(i, j - 1) * _
        arr_Node_Probabilities_trans(i, j - 1) + arr_Node_State_Prices(i - 1, j - 1) * (1 - arr_Node_Probabilities_trans(i - 1, j - 1))) / inf_no_dividend
Next i
arr_Node_State_Prices(0, j) = arr_Node_State_Prices(0, j - 1) * arr_Node_Probabilities_trans(0, j - 1) / inf_no_dividend
'-----
' Calculate the probabilities of reaching each node in the tree
'-----

```

```

For i = 0 To j ' vertical nodes
    arr_Node_Probabilities(i, j) = arr_Node_State_Prices(i, j) * inf_no_dividend ^ j ' from the definition of Arrow-Debreu state price
Next i
Next j
'-----
' Calculate the local volatilities implied by the implied binomial tree
' See : http://www.xplere-stat.de/help/IBTlocsigma.html for comparison
' See also page 152 Applied Quantitative Finance PDF Book, Equation (7.17)
' This implementation assumes that the temp node probability arr_Node_Probabilities_temp(i, j) = 1
' where (i,j) corresponds to the node (i.e. stock price) for which the implied
' volatility is being calculated i.e. arr_Stock_Prices(i, j)
'-----
If Steps <= 20 Then ' otherwise the size of the Local_impl_vol array becomes too large, see max_impl_vol_rows
    row_count = 0
    For j = 0 To Steps - 1 ' time steps
        For i = 0 To j ' vertical nodes
            time_to_ex = TimeStep ' used later in the output array
            ReDim arr_Node_Probabilities_temp(0 To Steps, 0 To Steps) ' this is required to clear the contenTimeStep of arr_Node_Probabilities_temp,
do not move it to top of function
            arr_Node_Probabilities_temp(i, j) = 1 ' we assume that the probability of being at arr_Stock_Prices(i, j) is 1
            For w = j + 1 To Steps ' the possible time steps (or poinTimeStep) at which local volatilities can be calculated
                ' Calculate the temporary node probabilities
                arr_Node_Probabilities_temp(i, w) = arr_Node_Probabilities_temp(i, w - 1) * arr_Node_Probabilities_trans(i, w - 1)
                For V = i + 1 To w - 1
                    arr_Node_Probabilities_temp(V, w) = arr_Node_Probabilities_temp(V, w - 1) * arr_Node_Probabilities_trans(V, w - 1) + _
                    arr_Node_Probabilities_temp(V - 1, w - 1) * (1 - arr_Node_Probabilities_trans(V - 1, w - 1))
                Next V
                arr_Node_Probabilities_temp(w, w) = arr_Node_Probabilities_temp(w - 1, w - 1) * (1 - arr_Node_Probabilities_trans(w - 1, w - 1))
                ' Calculate the expected value of the log stock price, Applied Quantitative Finance PDF Book, Equation (7.17)
                mean_log_price_sum = 0
                For V = i To w
                    mean_log_price_sum = mean_log_price_sum + arr_Node_Probabilities_temp(V, w) * Log(arr_Stock_Prices(V, w))
                Next V
            Next w
        Next i
    Next j

```

```

' Calculate the log variance, Applied Quantitative Finance PDF Book, Equation (7.17)
log_var_sum = 0
For V = i To w
    log_var_sum = log_var_sum + arr_Node_Probabilities_temp(V, w) * (Log(arr_Stock_Prices(V, w)) - mean_log_price_sum) ^ 2
Next V
' Write the calculated values into the output array
Local_impl_vol(row_count, 0) = arr_Stock_Prices(i, j) ' stock price
Local_impl_vol(row_count, 1) = time_to_ex ' time to expiry for calculation of the implied local volatility
Local_impl_vol(row_count, 2) = log_var_sum ^ 0.5 ' implied local volatility from variance
row_count = row_count + 1
time_to_ex = time_to_ex + TimeStep
Next w
Next i
Next j
End If ' Steps <= 20

Call_Sum = 0
Put_Sum = 0

For i = 0 To Steps ' terminal vertical nodes
    Call_Sum = Call_Sum + arr_Node_State_Prices(i, Steps) * Application.Max(arr_Stock_Prices(i, Steps) - Strik, 0)
    Put_Sum = Put_Sum + arr_Node_State_Prices(i, Steps) * Application.Max(Strik - arr_Stock_Prices(i, Steps), 0)
Next i

If Option_Type = 0 Then ' Call option
    Out_Opt_Price = Call_Sum
Else ' Put option
    Out_Opt_Price = Put_Sum
End If

DK = Out_Opt_Price
End Function

```

Function D.7: Vol_Smile

Function Vol_Smile(Underlier, Strik)

 Vol_Smile = 0.4 + (Underlier - Strik) / Underlier / 4

End Function