

# Modelling the implied volatility surface: an empirical study for FTSE options

Author: Amadeo Alentorn (aalent@essex.ac.uk)  
PhD in Computational Finance  
Centre of Computational Finance and Economic Agents  
University of Essex

Supervisors: Dr. Sheri Markose & Dr. Kyriakos Chourdakis

May 2004

## **Abstract**

The volatility surface implied by option prices presents a structure that changes over time. The aim of this study is to present a framework to model the implied volatility of the FTSE options in real time, and to present a prototype application that implements this framework. We adapt the parametric models presented in Dumas *et al* (1998) to estimate the surfaces across moneyness instead of across strikes. We discuss how this framework can be used in applications of option pricing and risk management.

Keywords: implied volatility surface, real time modelling, option pricing

JEL classification codes: G13

The latest version of this paper and the program can be downloaded from:

<http://privatewww.essex.ac.uk/~aalent/>

## **1. Introduction**

According to the classic Black-Scholes option pricing model (1973), all option prices on the same underlying asset with the same expiration date but with different exercise prices should have the same implied volatility, and therefore, plotted as a surface, it should be flat. However, this is not what is observed from traded option prices, and it is well documented that it exhibits a smile across strikes, and term structure across time to maturity (Cont and Fontseca, 2001). Regardless of the invalid constant volatility assumption of the Black-Scholes model, in practice, the implied volatility can be obtained by inverting the Black-Scholes formula given the market price of the option. As presented by Rebonato (1999), the implied volatility is the ‘wrong number to plug into the wrong formula to get the right price’. In order to obtain an implied volatility for any given strike and maturity, it is necessary to interpolate or smooth the discrete data from the few options actively traded. Derman & Kani (1994), Dupire (1994) and Rubinstein (1994) have presented models where the volatility of the underlying asset is a deterministic function of the asset price and time, and have developed a volatility function that has the potential of fitting the observed cross-section of option prices. More recent studies have used parametric models to fit the implied volatilities. Ncube (1996) estimated the FTSE 100 time-varying volatility using OLS regression on both time to maturity and strike price. The model depended on strike and the first and second powers of time to maturity. Similarly, Dumas *et al* (1998) proposed several volatility functions, as functions of both strike and time to expiration, for S&P 500 options. In a more recent study, Peña *et al* (1999) estimated the volatility function for the Spanish index IBEX-35 with six alternative structural forms, but only as a function of moneyness.

This paper is organized as follows: the next section contains a description of the model, which is based on the adaptation of the parametric models from Dumas *et al* (1998) to estimate the surfaces across moneyness instead of across strikes. The data are described in Section 3. The estimation results of the volatility surfaces and implied densities are discussed in Section 4. Section 5 presents the application specifications and some details of our implementation of the mode. In Section 6 we explain how this model can be used in the areas of option pricing and risk management. We finally conclude in Section 7 and propose future extensions to our model.

## 2. The model

### 2.1 Obtaining the of implied volatility with the Newton-Raphson algorithm

Recall that the Black-Scholes European call option formula is

$$C = S_0 e^{-qT} N(d_1) - K e^{-rT} N(d_2)$$

where  $S_0$  is the underlying asset (in our case, the FTSE 100 index),  $q$  is the expected dividends paid over the option's life,  $X$  is the option's strike price,  $T$  is the time to expiration,  $r$  is the risk-free interest rate,

$$d_1 = \frac{\ln(S_0 / K) + (r - q + \sigma^2 / 2)T}{\sigma \sqrt{T}} \quad d_2 = d_1 - \sigma \sqrt{T}$$

$\sigma$  is the volatility rate, and  $N(d)$  is the cumulative unit normal density function with upper integral limit  $d$ . The implied Black-Scholes volatility can be found uniquely from traded option prices because of the monotonicity of the Black-Scholes formula in the volatility parameter:

$$\frac{\partial BS}{\partial \sigma} > 0$$

The Newton-Raphson algorithm provides a numerical way to invert the Black-Scholes formula in order to recover  $\sigma$  from the market prices of the call option  $C$  (or Put option  $P$ )

$$f(\sigma) = BS(\sigma) - C = 0.$$

The Visual Basic implementation of this algorithm can be found in Appendix C. This algorithm implementation is more efficient than the function in the MatLab Financial toolbox, since it checks for arbitrage violations before starting the iterations. MatLab takes especially longer when the option price violates the arbitrage condition, that is, the price of the call is smaller than Black-Scholes price in the limit of zero volatility:

$$\lim_{\sigma \rightarrow 0} BS(\sigma) > C,$$

## 2.2 Implied dividend yield and interest rate

A stock index can be considered as the price of portfolio of stocks that pay dividends. If  $q$  is the dividend yield rate on the portfolio represented by the index during the life of the contract (expressed with continuous compounding), the equation for the futures price  $F_o$  is:

$$F_o = S_o e^{(r-q)T}$$

The put-call parity for an option on an index is equivalent to the one for an option on a stock paying dividend yield at rate  $q$ :

$$c + Xe^{-rT} = p + S_o e^{-qT}$$

Theoretically, the implied dividend yield  $q$  and interest rate  $r$  can be retrieved by simultaneously solving the above system of equations. In total, there are four prices that need to be used: a call option and a put option with the same strike, the futures contract with the same expiry date as the options, and the underlying index price,  $S_o$ . By using price triplets with different expiry dates, one would be able to construct the dividend yield and interest rate curve, and extrapolate from that curve the  $r$  and  $q$  values for any given time to maturity.

However, in practice it is rather difficult to obtain synchronized price triplets, since only one of the FTSE 100 futures contracts is heavily traded (the one closest to expiration), and only a small range of options are continuously traded (the options close to at the money, and close to expiration). Therefore, the interest rate and dividend yield cannot be obtained by using this approach. The values of these parameters quoted in DataStream will be used instead.

## 2.3 Construction of smooth volatility surfaces

As presented in the introduction, several studies fitted parametric volatility functions to observed implied volatilities. We are interested, as in Dumas *et al* (1998) in estimating the volatility across two dimensions: time to expiration and strike. However, according to Cont (2002), implied volatility patterns across moneyness vary less in time than when expressed as a function of the strike. Therefore, we will test the

models presented in Dumas *et al* (1998) in the FTSE 100 index, but modifying the parametric forms so they are a function of moneyness instead of a function of strike.

Also, there is an additional computational benefit on doing the regression on moneyness rather than on strike prices, since the function is of a simpler form, and therefore, the estimation algorithm converges faster.

We will study the following three models:

$$\text{Model 0: } \sigma(MN, T) = \beta_0 + \varepsilon$$

$$\text{Model 1: } \sigma(MN, T) = \beta_0 + \beta_1 MN + \beta_2 MN^2 + \varepsilon$$

$$\text{Model 2: } \sigma(MN, T) = \beta_0 + \beta_1 MN + \beta_2 MN^2 + \beta_3 T + \beta_4 T MN + \varepsilon$$

Model 0 is the volatility function representing the constant volatility model of Black-Scholes, that yields a constant volatility regardless of the level of moneyness and time to maturity. Model 1 attempts to capture the quadratic volatility smile across moneyness, and Model 2 captures additionally the variation across the time dimension, as well as a combined effect of time and moneyness. The moneyness is defined according to Gross and Waltner (1995) as  $MN = (\log(F / K)) / \sqrt{T}$ .

In principle, call and put options should yield the same implied volatility, based on the Call Put parity theorem. Some argue (Ncube, 1996) that since the put option is a natural hedging instrument, investors may be willing to pay more for it, and therefore, its implied volatility would be higher than the call counterpart. However, we will not take into account this possible bias in our model, since we are interested in obtaining a more generic solution, from which both call and put options can be priced.

The interpretation of the coefficients in this model is as follows. The parameter  $\beta_0$  present in all models is the constant of the regression. In Models 1 and 2, the smile across moneyness is captured by a second order function that represents the parabolic shape of the smile. The  $\beta_1$  coefficient controls the displacement of the origin of the parabola with respect to the at-the-money (ATM) options, while the  $\beta_2$  coefficient controls the wideness of the smile. In Model 2, the additional coefficients  $\beta_3$  and  $\beta_4$  capture the effect of time to expiration, i.e. the term structure of the implied volatility.

## 2.4 Optimization procedure

To estimate the parameter vector  $\beta$ , which represents the coefficients of the models, the nonlinear least-squares function in *MatLab* is used. The algorithm in this function minimizes the *sum of squares error* (SSE) between observations  $y_i$  and the model's prediction for the  $i$ -th observation  $prd_i(w)$  for a given parameter vector  $\beta$ :

$$\min SSE(\beta) = \sum_{i=1}^m (y_i - prd_i(w))^2$$

where  $prd_i(w)$  is the user-defined function that implements each specific model of the volatility surface equation. This statistic measures the total deviation of the observations from the estimated surface. To examine the goodness of fit statistics, we can then calculate the  $R^2$  statistic, which is a measure of how successful the fit is in explaining the variation of the data, that is, it measures the fraction of the variance in the data that is explained by the regression. The  $R^2$  statistic, for the nonlinear least-squares estimation, is defined as the square of the correlation between the observations (Greene, 2000:420), and is calculated with the following equation:

$$R^2 = 1 - \frac{SSE(\beta)}{\sum_{i=1}^m (y_i - \bar{y})^2}$$

## **3. The data**

The data set used to calibrate the volatility surfaces is comprised of option prices, futures prices and the FTSE 100 index for five days, between 19<sup>th</sup> of April to 23<sup>rd</sup> of April 2004. Each price was time-stamped with the exact time when the transaction took place, and therefore, we were able to match the option prices with the underlying price at the time the option was traded, in order to calculate the individual implied volatilities. The results we report in this study are for end of day prices, even though our application is able to estimate the volatility surface in real time during the trading day. Very few options were continuously traded, so at the end of the day, the latest price for most options is from sometime in the middle of the day. There were, in average, 100 different option contracts traded every day, with the ones close to maturity and at the money being traded most frequently.

## 4. Results

### 4.1 Estimated volatility surfaces

Each of the three models was estimated for each of the five days in the data set, using the estimator described in the section 2. Table 1 below shows the estimated constant parameter for **Model 0**, and Figure 1 below shows the implied volatility surface for 19<sup>th</sup> April 2004. As expected, the volatility surface is a simple flat surface. Nevertheless, this benchmark model will be used for comparison purposes.

Date	Parameters $\beta_0$	RMSE (0.0786)	R <sup>2</sup>
19-Apr-04	0.1236	0.0306 (0.1139)	0
20-Apr-04	0.1205	0.0328 (0.1589)	0
21-Apr-04	0.1307	0.0375 (0.0952)	0
22-Apr-04	0.1247	0.0309 (0.1088)	0
23-Apr-04	0.1214	0.0340 (0.1088)	0

Table 1: Estimated parameters for Model 0, average root mean squared errors RMSE (with value of the squared 2-norm of the residual in parentheses), and R<sup>2</sup> statistic.

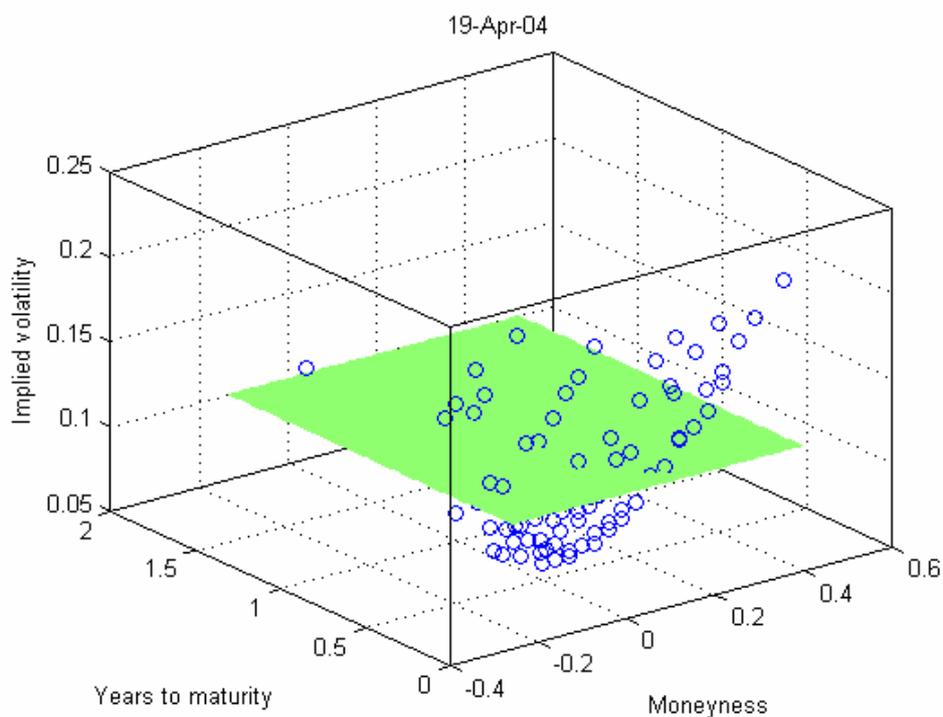


Figure 1: Estimated implied volatility surface for Model 0 on 19th April 2004. The blue circles represent the observed implied volatilities.

The estimation results for **Model 1** can be found in Table 2 below. When comparing the RMSE values of Model 0 with Model 1, we can see that Model 1, with an RMSE of 1.63%, is a considerable improvement over Model 0, which had an average RMSE of 3.33%, by more than halving the RMSE statistic. The RMSE values are for an average implied volatility of 12.42%. If we look at the goodness of fit of Model 1 with the  $R^2$  statistic, the model has an average value of 0.8637, that is, it is able to explain 75.25% of the variance on the implied volatility. The benchmark model has an  $R^2$  of zero, since it is a constant function.

Date	Parameters			RMSE	$R^2$
	$\beta_0$	$\beta_1$	$\beta_2$		
19-Apr-04	0.1122	0.1341	0.4235	0.0138 (0.0159)	0.7976
20-Apr-04	0.1105	0.1927	0.1421	0.0144 (0.0218)	0.8084
21-Apr-04	0.1162	0.2003	0.1479	0.0178 (0.0359)	0.7742
22-Apr-04	0.1132	0.1001	0.3381	0.0188 (0.0355)	0.6272
23-Apr-04	0.108	0.1802	0.1271	0.0168 (0.0266)	0.7551

Table 2: Estimated parameters for Model 1, average root mean squared errors RMSE (with value of the squared 2-norm of the residual in parentheses), and  $R^2$  statistic.

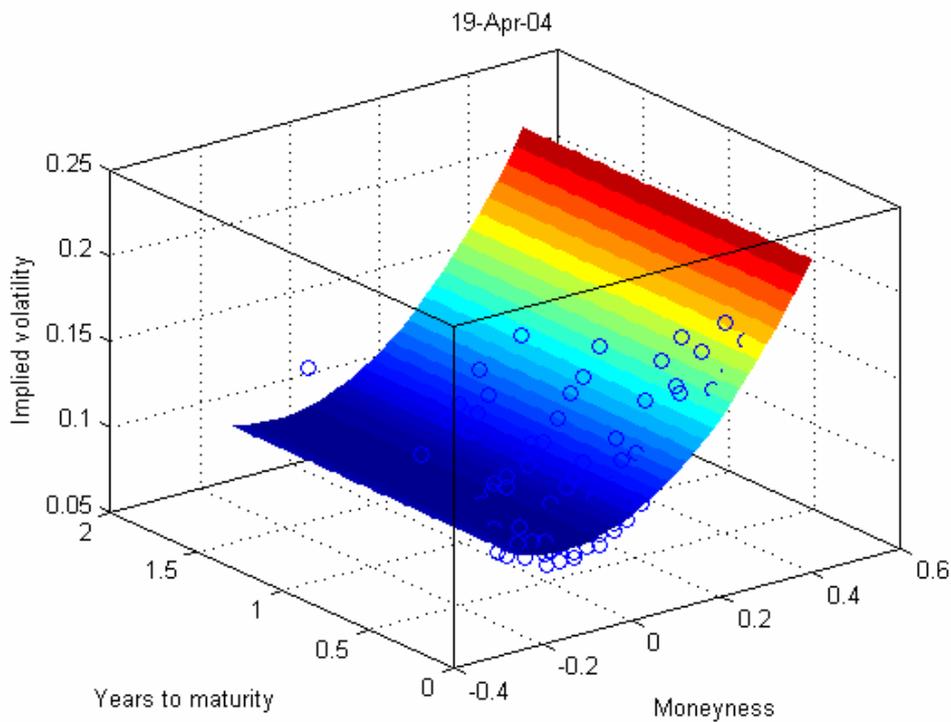


Figure 2: Estimated implied volatility surface for Model 1 on 19th April 2004. The blue circles represent the observed implied volatilities.

And finally, the estimated parameters for **Model 2**, and a sample implied volatility surface, can be found in Table 3 and Figure 2 below. The implied volatility surface corresponding to Model 2 for each of the 5 days can be found in Appendix A. The average RMSE for Model 2 is 1.11%, with an average  $R^2$  statistic of 88.33%. This model, with two additional parameters to capture the term structure dependency, results on a 30% reduction of the RMSE with respect to Model 1, and on a 13% improvement on explaining the variance of the volatility.

Date	Parameters					RMSE	$R^2$
	$\beta_0$	$\beta_1$	$\beta_2$	$\beta_3$	$\beta_4$		
19-Apr-04	0.096	0.0675	0.5961	0.0374	0.2599	0.0083 (0.0058)	0.9261
20-Apr-04	0.0952	0.1257	0.3576	0.0333	0.2039	0.0082 (0.0071)	0.9379
21-Apr-04	0.102	0.1418	0.3307	0.0284	0.1836	0.0132 (0.0197)	0.8763
22-Apr-04	0.099	0.0408	0.4671	0.0272	0.2762	0.0140 (0.0195)	0.7955
23-Apr-04	0.0951	0.1345	0.3098	0.023	0.1138	0.0117 (0.0129)	0.8810

Table 3: Estimated parameters for Model 1, average root mean squared errors RMSE (with value of the squared 2-norm of the residual in parentheses), and  $R^2$  statistic.

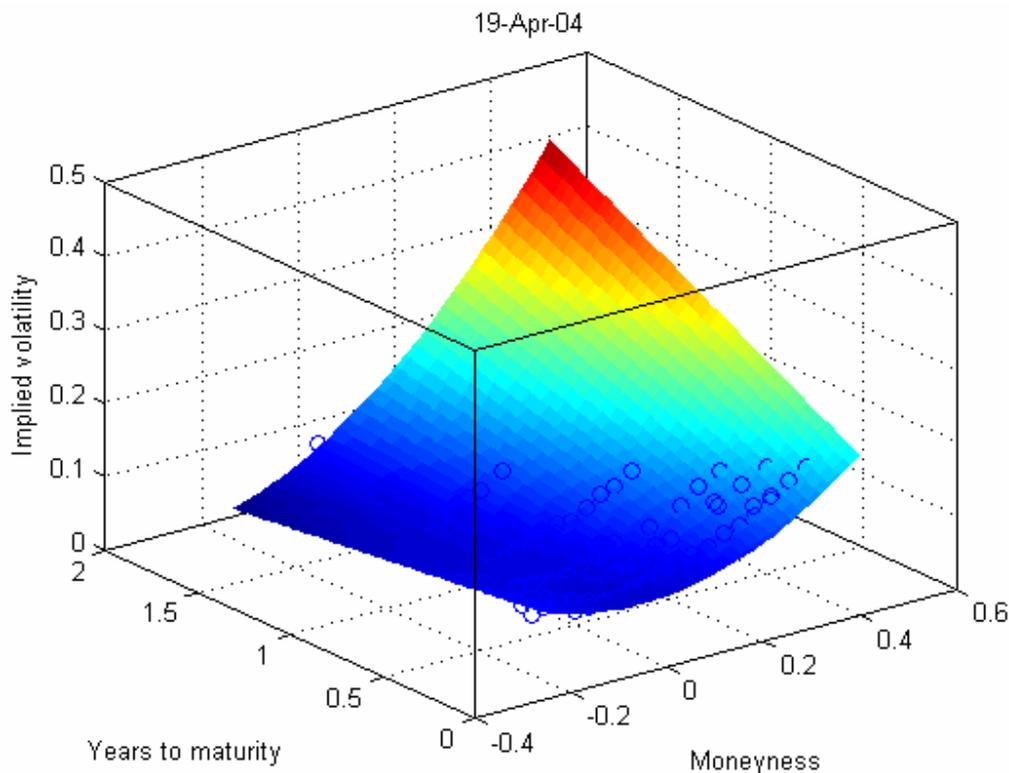


Figure 3: Estimated implied volatility surface for Model 2 on 19th April 2004. The blue circles represent the observed implied volatilities.

## 4.2 The implied probability density function

We now look at the corresponding risk-neutral probability density function by using the Breeden and Litzenberger (1978) result, which gives the relationship between the European option prices and the risk-neutral probability distribution. This result states that the second derivative of the European call price function  $C$  taken with respect to its strike price  $K$  is equal to the continuously discounted risk-neutral distribution  $f^Q(K)$  for the price of the underlying at maturity

$$\frac{\partial^2 C}{\partial K^2} = e^{-rT} f^Q(K)$$

where  $T$  is the remaining time to maturity. Since we have a closed form solution for the volatility (as a function of moneyness and time to expiration), we can compute this derivative analytically (see Appendix D for the *Mathematica* notebook)

There is only one FTSE 100 futures contract that is heavily traded in LIFFE, and therefore, we can only use the put-call parity relationship for options that expire in the same day as the closest futures contract. The futures contract closer to expiration was the 18<sup>th</sup> June 2004 contract. We only calculate the density function for Model 2, since it was the one that best fitted the data. The density function, plotted across all strike prices, becomes negative for strikes lower than 1600. However, the area of interest is the one for which the option prices are available. In our data set, this range is from 4000 to 5000. Therefore, the density function will be truncated, and only the relevant section, highlighted in a dashed rectangle in Figure 4, will be used.

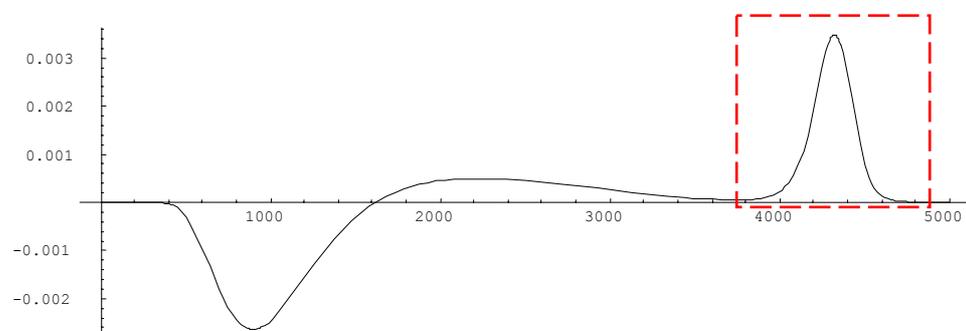


Figure 4: implied density function for Model 2 across all strikes.

To ensure that it is a consistent probability measure, we must enforce that the probabilities add up to 1, that is,  $\int_{K_{\min}}^{K_{\max}} f(K) = 1$ . We can enforce this constraint by dividing the density function by the area under the curve, in order to normalize it. Figure 5 shows the final implied density function, after being truncated and normalized.

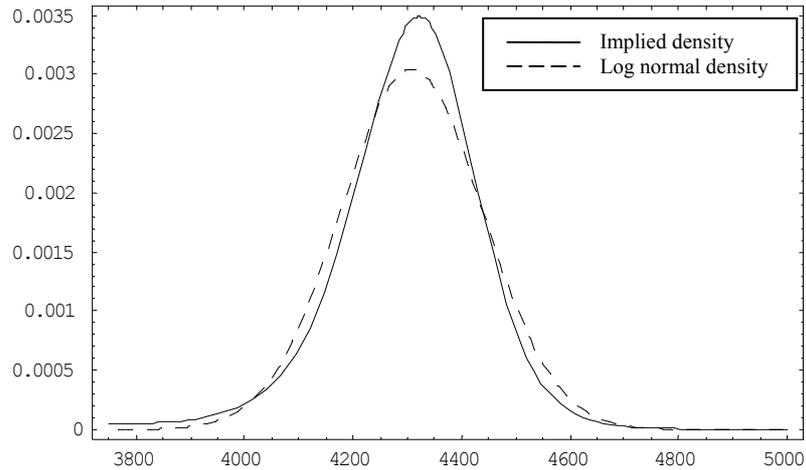


Figure 5: Implied density versus normal density on 19<sup>th</sup> April 2004

The implied density function satisfies the risk neutral constraint since its mean is 4308.07 approximately equal to the futures price, 4303.58, with an error of 0.1%. We can observe the classic stylized facts when comparing the implied density with its equivalent normal distribution (the dashed line): the implied density has a fatter left tails than the lognormal distribution, i.e. the implied distribution is more leptokurtic. Additionally, the distribution is skewed to the right.

## **5. Design and Implementation**

This section aims to give a very general outline of the application developed to implement the model. The technical details regarding the implementation and integration of the different components can be made available by the author on request. Selected portions of the source code are included in the Appendices.

### **5.1 Application specification**

The functional specification for an application to implement the model in this paper can be summarized in the following five points:

1. Download of pricing data with time-stamps in real-time, to ensure synchronized prices between options, and underlying and futures.
2. Storage of prices in a local database for future reference.
3. Calculation of implied volatilities from option prices.
4. Estimation of implied volatility surfaces for a variety of models.
5. Plotting graphs and compiling statistics.

### **5.2 Implementation**

A menu driven application with a graphical user interface that implemented the requirements presented above was developed as part of this project. The core of the application was developed in *Visual Basic 6*, but it was integrated with the following components/environments:

- *Microsoft WebBrowser* control and *Java RunTime* to connect and download real-time pricing data from the [ww.liffe-data.com](http://ww.liffe-data.com) website (quoted prices are delayed by 15 minutes).
- A database was designed in *MS Access* to store the pricing data for options, futures and the index.
- Integration with *MatLab* via *COM* permitted access the *Optimization Toolbox*, running custom *MatLab* programs to estimate the volatility surfaces, and to generate 3D graphs.

The application automatically downloads prices from the LIFFE website via the internet, and estimates the implied volatility surface in real time.

## **6. Applications**

There are two main areas in finance where a real time application that calculates the implied volatility surface would be useful. The first one is for live pricing of exotic options on a trading desk, where exotic options need to be valued in consistency with the volatility implied by liquid traded options. The second area is risk management.

### **6.1 Pricing path dependant exotic options with implied trees**

One application where implied volatility surfaces can be used in when constructing implied trees to price path dependant exotic options. The construction of implied binomial trees has been outlined by Derman and Kani (1994). In an implied binomial tree, option prices for all strikes and expirations, obtained by interpolation from the volatility surface, determine the position and probability of reaching each node. This model, even though it allows the pricing of exotic contracts consistently with known prices of traded vanilla options, runs into the problem of negative probabilities. There are several algorithms to “fix” the negative probabilities, but the model becomes very restrictive. A later model of implied trinomial trees was presented by Derman *et al* (1996), which avoids the negative probabilities problem by assigning an appropriate state space, and thus, only having to calculate the probabilities and Arrow-Debreu prices at each node. Regardless of which approach is used, having an accurate volatility surface that reflects the traded option prices is essential in order to construct a consistent tree and accurately price path dependant exotic options.

### **6.2 Risk management applications: Value-at-Risk Estimation**

In risk management, one of the most widely used methods is Value-at-Risk (VaR). The VaR of a given portfolio summarizes the expected maximum loss over a target horizon within a given confidence interval. The implied volatility models estimated in section 4 can be used to perform VaR calculations for portfolios containing assets whose payoffs /returns depend on the FTSE index, since the implied volatilities represent an unbiased and efficient forecast of the future, average volatility (Cassese and Guidolin, 2003). The portfolio can contain any combination of equity shares, vanilla options and exotic options, and worst case future scenarios can be simulated using the implied volatility surface.

## **7. Conclusions and future work**

We have estimated three different models to the observed implied volatilities for the FTSE 100 options. The model that best fitted the data was Model 2, which captured both the smile across moneyness, as well as the term structure across time to expiration.

A prototype application has been developed to implement the model described in this paper, in real-time. The estimation of the volatility surface for the parametric models is fast enough (around 2 seconds) to be done in real time. At this stage, our application calibrates the entire surface for each new observation. However, a Kalman filter approach could be implemented to speed up this process and use each new observation to update the estimated parameter vector.

From the data collected, we observed that only a very small set of options are actively traded, and those are the contracts that are both ATM and close to expirations. As a consequence, at a given time during the day, we may only have recent prices for a small subset of options. The last traded price for options deep in of the money is usually from several days ago, and for options in the money, the last traded price is from today, but from several hours ago.

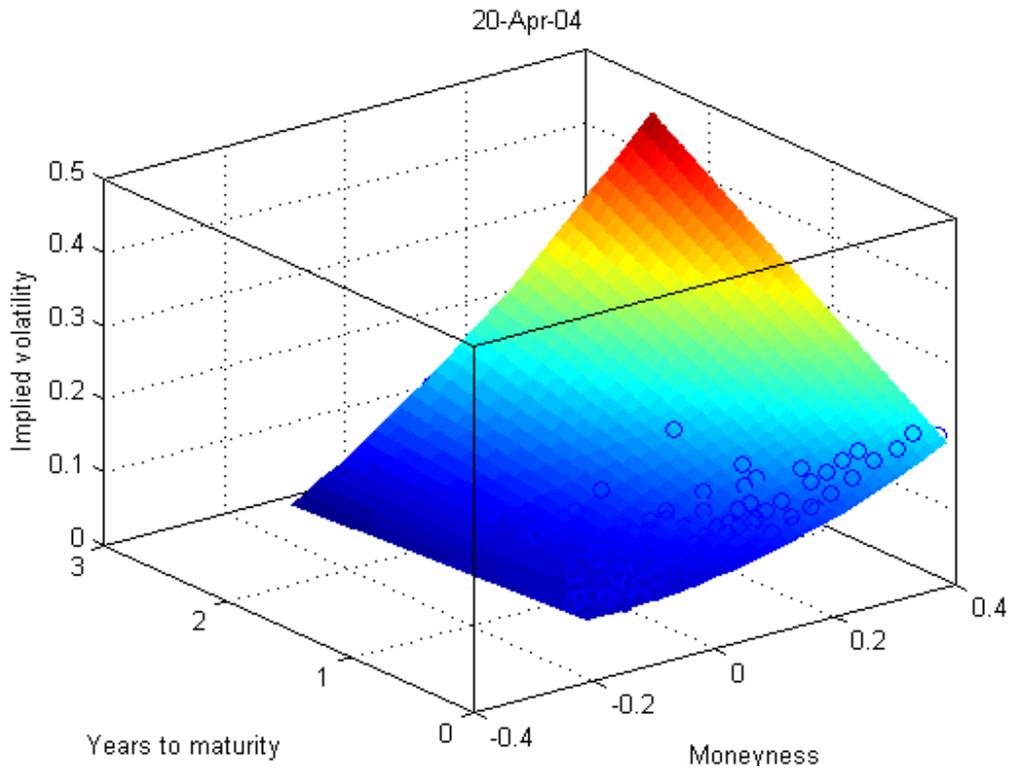
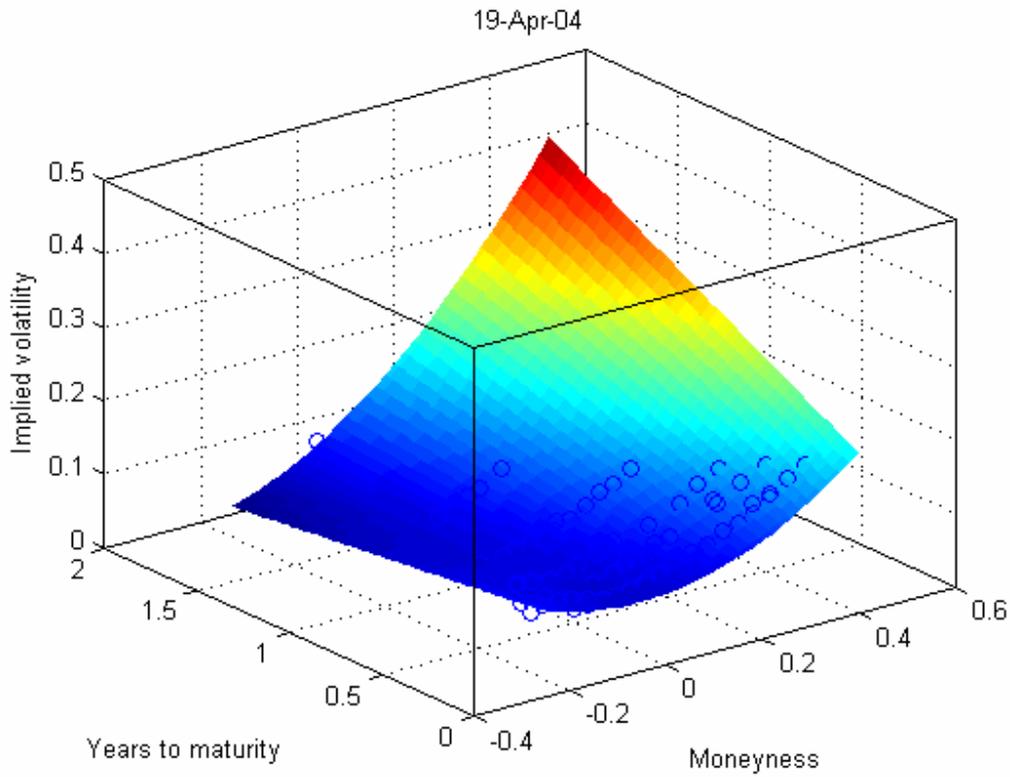
Therefore, in order to obtain a more accurate estimate of the current market expectation of the volatility, a future study could modify the minimization problem, so it uses a weighted sum of squares errors, where the weights would be inversely proportional of the time elapsed from the last observation, and thus, giving more relevance to recent observations. The older observations would still contribute to the overall estimation, but would have a smaller impact in defining the surface. The regression could minimize the following equation, where  $\tau_i$  is the time elapsed since the option  $i$  was traded:

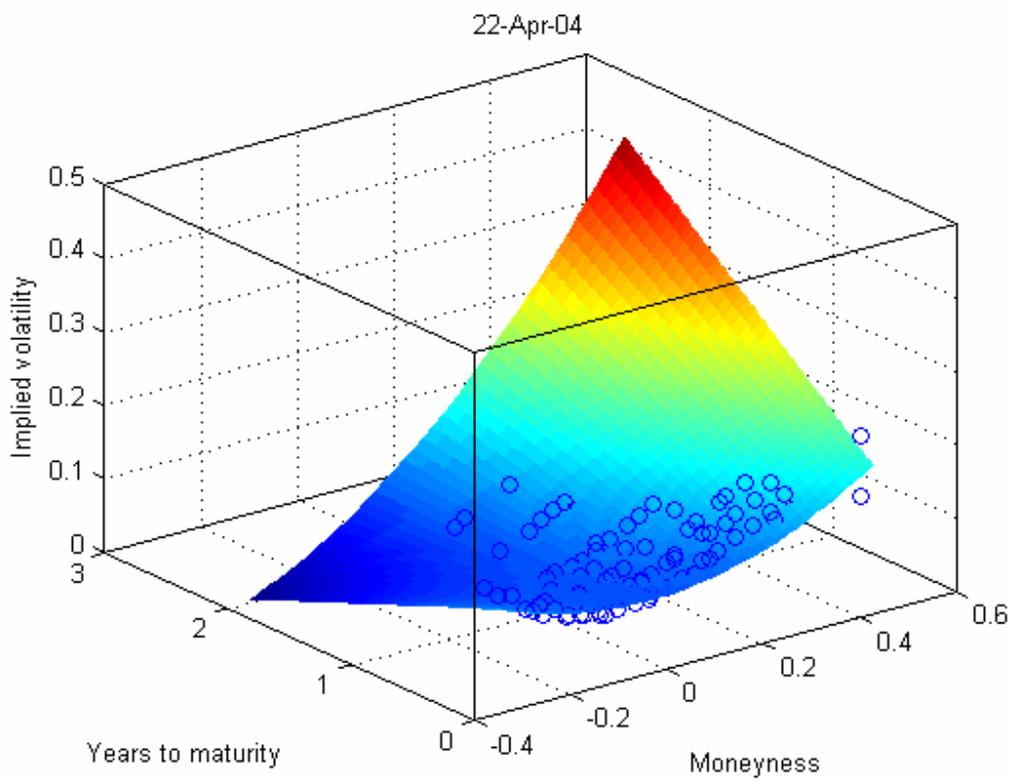
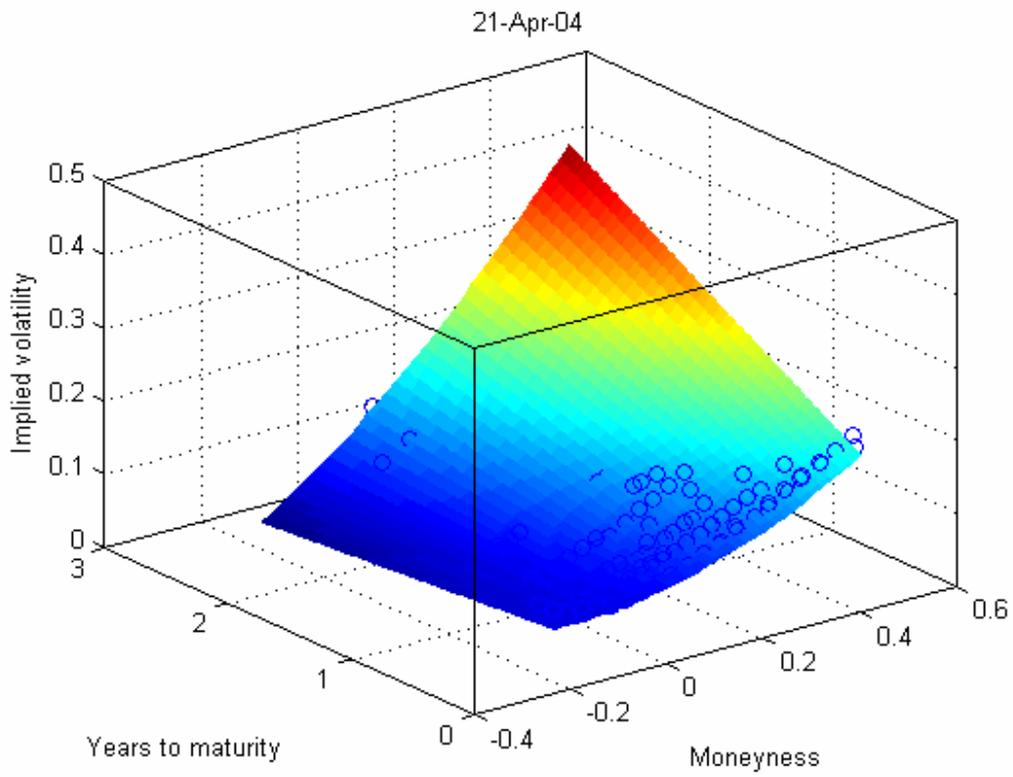
$$\min SSE(\beta) = \sum_{i=1}^m \frac{1}{(1 + \tau_i)} (y_i - \text{prd}_i(w))^2$$

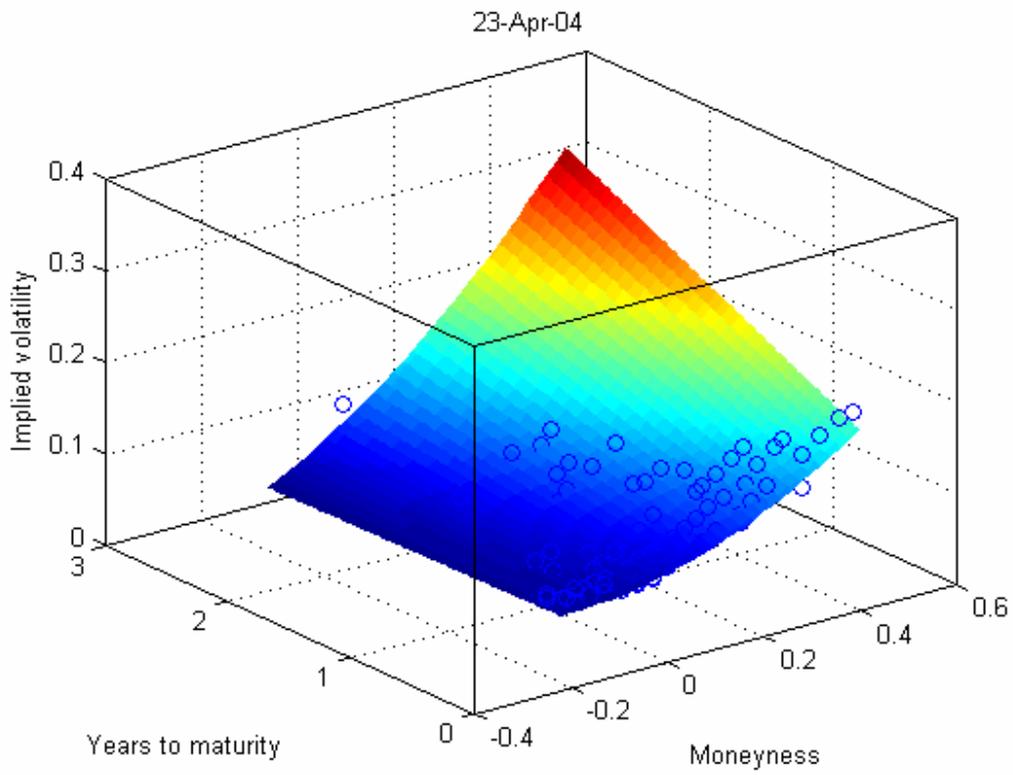
## **8. References**

- Black, F., and M. Scholes (1973) “The Pricing of Options and Corporate Liabilities“  
*Journal of Political Economy*, 81, 637-654
- Breeden, D. T. and Litzenberger, R. H. (1978), “Price of state-contingent claims implicit in option prices”, *Journal of Business* 51 no. 4, 621--651.
- Cassese, G. and Guidolin, M. (2003) “Modelling the Implied Volatility Surface: Does Market Efficiency matter?”, *University of Virginia working paper*
- Cont R. and Fonseca J.d. (2002) “Dynamics of implied volatility surfaces”,  
*Quantitative Finance* 2 (February 2002) 45-60
- Derman, E., and Iraj K., (1994) “The volatility smile and its implied tree”,  
*Quantitative Strategies Research Notes*, Goldman Sachs
- Derman, E., Kani, I. Chriss, N. (1996) “Implied Trinomial Trees of the Volatility Smile”, *Quantitative Strategies Research Notes*, Goldman Sachs
- Dumas, B., Fleming J. and Whaley R.E. “Implied Volatility Functions: Empirical Tests (1998), *The Journal of Finance*, Vol LIII, No. 6, 2059-2106
- Dupire B., (1994), “Pricing with a smile”, *Risk* 7, 18-20
- Greene, W. (2000) “Econometric Analysis”. Fourth Edition. *Prentice-Hall, Inc.*
- Gross, L., and N. Waltner, (1995) “S&P 500 Options: Put Volatility Smile and Risk Aversion”, *Salomon Brothers, mimeo*
- Ncube M. (1996) “Modelling implied volatility with OLS and panel data models”,  
*Journal of Banking and Finance* 20 71-84
- Peña, I., Serna, G, and G. Rubio (1999). “Why do we smile? On the determinants of the implied volatility function”, *Journal of Banking and Finance* 23 1151-1179
- Rebonato R. (1999) ‘Volatility and Correlation’, *John Wiley*, Chichester
- Rubinstein M., (1994), “Implied binomial trees”, *Journal of finance*, 49, 771-818
- Syrdal S.A. (2002) “A Study of Implied Risk-Neutral Density Functions in the Norwegian Option Market”, *Norges Bank Working Paper*, 2002/13

**Appendix A: Implied volatility surfaces for Model 2**







## **Appendix B: Newton-Raphson algorithm implementation in Visual Basic**

*Adapted from a C++ implementation at:*

*[http://finance.bi.no/~bernt/gcc\\_prog/algoritms\\_v1/algoritms/node8.html](http://finance.bi.no/~bernt/gcc_prog/algoritms_v1/algoritms/node8.html)*

**Public Function calculate\_implied\_volatility\_bisections( \_  
s As Double, X As Double, r As Double, t As Double, option\_price As Double, d As  
Double, sType As String) As Double**

```
'check for arbitrage violations:
'if price at almost zero volatility greater than price, return 0

Dim sigma_low As Double
Dim Price As Double
Dim IV As Double
sigma_low = 0.0001

IV = -1

'calculate the price of the volatility was almost zero, to see what would be the minimum
price
Select Case sType
    Case "C": Price = option_price_call_black_scholes(s, X, r, sigma_low, t, d)
    Case "P": Price = option_price_put_black_scholes(s, X, r, sigma_low, t, d)
End Select

If Price > option_price Then
    IV = 0
Else
    'simple binomial search for the implied volatility.
    'relies on the value of the option increasing in volatility
    Const ACCURACY = 0.00001 'make this smaller for higher accuracy
    Const MAX_ITERATIONS = 100
    Const HIGH_VALUE = 10000000000#
    Const ERROR = -1E+40

    'want to bracket sigma. first find a maximum sigma by finding a sigma
    'with a estimated price higher than the actual price.
    Dim sigma_high As Double
    sigma_high = 0.3
    Select Case sType
        Case "C": Price = option_price_call_black_scholes(s, X, r, sigma_high, t, d)
        Case "P": Price = option_price_put_black_scholes(s, X, r, sigma_high, t, d)
    End Select

    Do While (Price < option_price)
        sigma_high = 2# * sigma_high 'keep doubling.
        Select Case sType
            Case "C": Price = option_price_call_black_scholes(s, X, r, sigma_high, t, d)
            Case "P": Price = option_price_put_black_scholes(s, X, r, sigma_high, t, d)
        End Select
        If (sigma_high > HIGH_VALUE) Then
            GoTo ReturnValue 'return ERROR; // panic, something wrong.
        End If
    End While
End If
```

```

Loop
Dim i As Integer
For i = 0 To MAX_ITERATIONS
    Dim sigma As Double
    sigma = (sigma_low + sigma_high) * 0.5
    Select Case sType
        Case "C": Price = option_price_call_black_scholes(s, X, r, sigma, t, d)
        Case "P": Price = option_price_put_black_scholes(s, X, r, sigma, t, d)
    End Select

    Dim test As Double
    test = (Price - option_price)
    If (Abs(test) < ACCURACY) Then
        IV = sigma
        Exit For
    ElseIf test < 0# Then
        sigma_low = sigma
    Else
        sigma_high = sigma
    End If
Next i
End If

Return Value:

If IV >= 0 Then
    calculate_implied_volatility_bisections = IV
Else
    'return ERROR;
    'Err.Raise 702
    calculate_implied_volatility_bisections = -1 'IV = -1000000
End If

End Function

```

## Appendix C: Mathematica notebook to calculate the implied density function

In[39]:=

**Quit[];**

In[1]:=

```
<< Statistics`ContinuousDistributions`
pdfn[y_] = PDF[NormalDistribution[0, 1], y];
nd[d_] := CDF[NormalDistribution[0, 1], d];
```

In[4]:=

$$d1 := \frac{\text{Log}[S / K] + (r + v^2 / 2) T}{v \sqrt{T}};$$

$$d2 := d1 - v \sqrt{T};$$

In[6]:=

$$\text{Cbs}[K_] = S e^{-qT} \text{nd}[d1] - K e^{-rT} \text{nd}[d2];$$

In[7]:=

```
S = 4300;
r = 0.04;
T = 1 / 12;
q = 0.03;
F = S e^{(0.04 - 0.03) T};
```

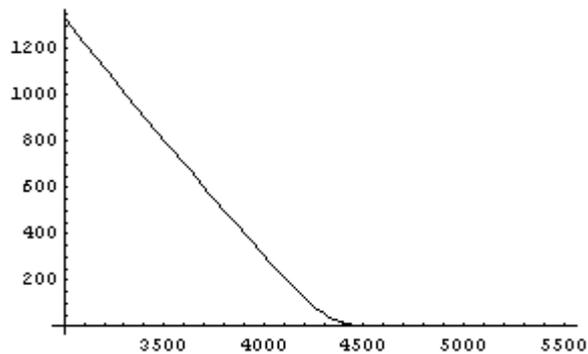
In[12]:=

$$Mn := \frac{\text{Log}[S / K]}{\sqrt{T}};$$

$$v := 0.43 * (Mn^2) * E^{(0 * T)} + 0.09 * Mn + 0.1 * E^{(0.22 * T)}$$

In[14]:=

```
Plot[Evaluate[Cbs[K]], {K, 3000, 5500}, PlotRange -> All];
```



In[15]:=

In[29]:=

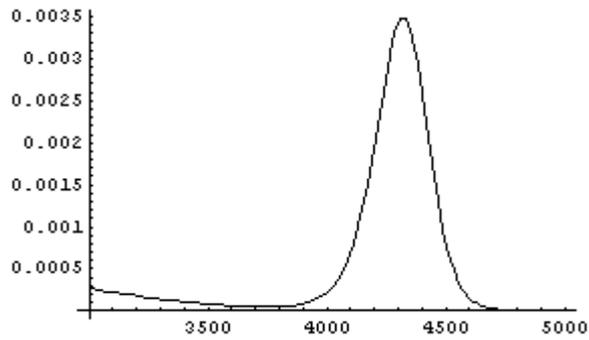
```
dns = D[Cbs[K], {K, 2}]
```

Out[29]:=

$$\frac{0.000160058 e^{-\frac{1}{2} \left( -4968.38 + 0.000201273 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right) \right)^2}}{K} + 2144.63 \left( \frac{0.000160593 e^{-2 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right)^2}}{K^2} - \frac{6.50574 \times 10^{-12} e^{-2 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right)^2} \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right)}{K^2} \right) - 0.498336 K \left( \frac{0.000160593 e^{-\frac{1}{2} \left( -4968.38 + 0.000201273 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right) \right)^2}}{K^2} - \frac{3.2323 \times 10^{-12} e^{-\frac{1}{2} \left( -4968.38 + 0.000201273 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right) \right)^2} \left( -4968.38 + 0.000201273 \left( 1.23424 \times 10^7 + \text{Log} \left[ \frac{4300}{K} \right] \right) \right)}{K^2} \right)$$

In[16]:=

```
Plot[dns, {K, 3000, 5000}];
```



In[17]:

```
NIntegrate[dns, {K, 3750, 5000}]
```

Out[17]:

```
1.00097
```

In[18]:

```
m = NIntegrate[K dns, {K, 3750, 5000}]
```

Out[18]:

```
4308.07
```

In[19]:

```
v = NIntegrate[(m - K)2 dns, {K, 3750, 5000}]
```

Out[19]:

```
17211.
```

In[20]:

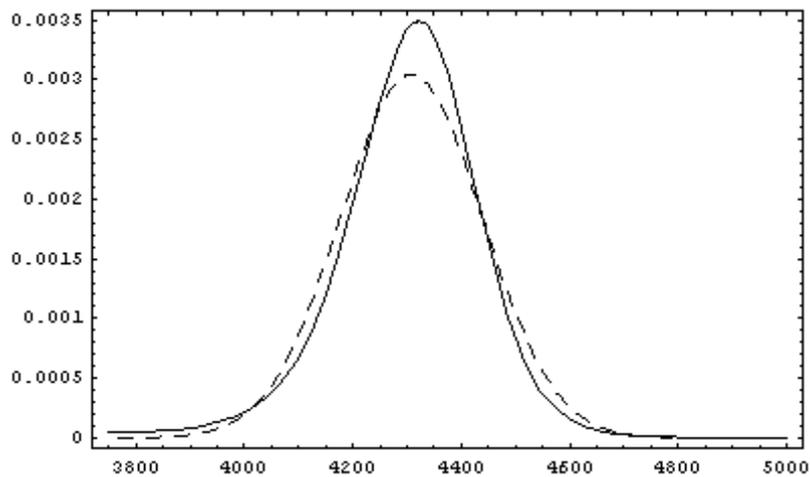
```
sd =  $\sqrt{v}$ 
```

Out[20]:

```
131.191
```

In[27]:

```
Plot[{dns, PDF[NormalDistribution[m, sd], K]}, {K, 3750, 5000}, PlotStyle → {RGBColor[0, 0, 0], Dashing[ {.02, .02}]}, Axes → False, Frame → True];
```



In[28]:

```
F
```

Out[28]:

```
4303.58
```

## Appendix D: Matlab code for estimation of volatility surfaces

*%Original code by Kyriakos Chourdakis (www.theonlytail.net)*

```
function [xxx,sss,rrr,R2,residuals,resnorms] = estimateVolSurf
(pts,WinTitle)
% clear;
format compact;

K = pts(:,1);
F = pts(:,2);
C = pts(:,3);
T = pts(:,4)/252;
IV = pts(:,5);
r = 0.03;
VM = mean(IV)
% fit volatility surface
MN = mness(K, F, T);
%pars = [ 1, -1, 0, VM, 1];
pars = [0.25, -1, 0.17, 0.10,0.20];
opts = optimset('Display','iter','TolFun',0.000000001);
[x,resnorm,residual] = lsqnonlin(@iverrors,pars,[], [], opts, MN, T,
IV)

%x =pars;
% display RMSE of parameter vector
e = iverrors(x,MN,T,IV);
STDERRORS = std(e);

R_square = 1 - (sum(e.^2))/(sum((IV - mean(IV)).^2))
RMSE = sqrt(mean(e.^2))

figure('Name',WinTitle,'NumberTitle','off'); %,'Box','on');

set(gcf,'Color','w');
%figure(1);
plot3(MN,T,IV,'o'); hold on;
%[DM,TM] = meshgrid(-.2:.02:.6,0:.02:1.2);
[DM,TM] = meshgrid(min(MN):.02:max(MN),0:.02:max(T));
ZM = ivfitted(x, DM, TM);
surf(DM,TM,ZM,'FaceColor','flat','EdgeColor','none'); hold off;
%mesh(DM, TM, ZM); hold off;
xlabel('Moneyness');
ylabel('Years to maturity');
zlabel('Implied volatility');
title(WinTitle);

%show axis and grids
box on;
grid on;

xxx= x;
sss = RMSE;
rrr = STDERRORS;
R2 = R_square;
residuals = residual
resnorms = resnorm
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y = ivfitted(pars, MN, T)
% The parametrized fitted surface

%Connstant volatility model
%y = pars(1) + MN.*0;

%Dumas Model 1
%y = pars(1) + pars(2)*MN + pars(3)*MN.^2;

%Dumas Model 2
%y = pars(1) + pars(2)*MN + pars(3)*MN.^2 + pars(4)*T + pars(5) * T
.* MN;

function y = iverrors(pars, MN, T, IV)
% The parametrization errors
y = ivfitted(pars, MN, T) - IV;

function y = mness(K, F, T)
% The moneyness function
y = log(F./K) ./ sqrt(T);

```