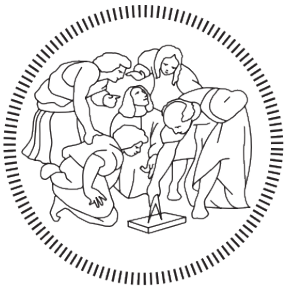


POLITECNICO
MILANO 1863

Financial Engineering 2023/24

Assignment 5 - Group 20

Margherita Bencini, Alessandro Torazzi, Matteo Torba, Giovanni Urso

Contents

1 Certificate Pricing

The first exercise concerned the pricing of a certificate issued by an hypothetical bank on February the 15th, 2008 at 10:45 C.E.T.in a single-curve interest rate setting. Other than the rate dynamics, we were able to neglect also the counterparty risk. Considering the Contract Conditions that we were provided, the only missing data was the participation coefficient alpha. We started by bootstrapping in order to compute the discount factors in the needed dates. The idea then was to impose the NPV of the two parties, A and B, to be equal to 0.

For party B, we had to simulate the asset dynamic (GBM) and we chose to implement the Antithetic Monte Carlo over the standard one in order to reduce the variance. We chose the number of simulations N = 10^5. This allowed us to compute the coupon paid by B at maturity, with the given formulas:

$$Coupon = \alpha(S(t) - K)^+$$

$$S(t) = \sum_{s=1}^{4} \sum_{n=1}^{2} W_n \frac{E_s^n}{E_{s-1}^n} \qquad with\ W_n = \frac{1}{8}$$

With this, we are able to finally Compute NPV_b in function of alpha.

The party A instead receives the 3 month Euribor plus the S^{spol}. For the computation of these discounted cash flows we used a "shortcut" and considered all the Euribor flows as 1 at the beginning minus one at the end of the time frame. Subtracting to the payments leg the 2% upfront, we were able to compute NPV_a.

Finally, imposing a null NPV of the contract, we obtained the value of alpha:

$$\alpha = \frac{(NPV_a - Upfront)}{\frac{1}{N}\sum_{i=1}^{N}(S_i(t) - K)^+} = 3.350132$$

With the following 95% confidence interval:

$$IC_\alpha = [3.350130, 3.350133]$$

2 Pricing Digital option

We were asked to verify the difference between the price of a digital option computed according to the Black model and in the case where one takes into account the smile in the curve of the implied

volatility. From theory we knew the price of a digital call per Black model:

$$dc(K) = B(t_0, t)N(d_2) = -\frac{d}{dK}c(K)$$

In order to take into account the implied volatility we priced the digital call through a call spread with notional equal to said spread: when making the spread tend to 0 we obtained:

$$dc = \lim_{\epsilon \to \inf} \frac{1}{\epsilon}[c(K, \sigma(K)) - c(K + \epsilon, \sigma(K + \epsilon))] = -\frac{d}{dK}c(K, \sigma(K))$$

Deriving then the Call price:

$$dc = -\frac{d}{dK}c(K, \sigma(K)) - (\frac{d}{dK}\sigma(K))(\frac{d}{d\sigma}c(K, \sigma))$$

We recognized the first term as the Black price for the digital call, which left the second term as the actual difference requested. We observed that $\frac{d}{dK}\sigma(K)$ could be considered as the slope of the line tangent to the volatility smile in K:

$$\text{slope} = \frac{\sigma_2 - \sigma_1}{K_2 - K_1}$$

with (K_1, K_2) the strike values, among the ones given, closest to K, and (σ_1, σ_2) their relative implied Volatilities.

$\frac{d}{d\sigma}c(K, \sigma)$ was clearly the Vega of a Call, for which we were missing only the volatility value that we decided to linearly interpolate using Matlab function `interp1` and considering again (K_1, σ_1) and (K_2, σ_2).

Finally we computed the difference as the product of slope, vega, notional and digital payoff, obtaining

$$\Delta dc(K) = 75882.9389$$

3 Pricing Call option

3.1 Case of $\alpha = 1/2$

We were asked to compute call prices on the 15^{th} of January 2008 at 10:45 C.E.T., according to a normal mean-variance mixture with $\alpha = 1/2$ model with given parameters of σ, κ and ν, and values of moneyness x between -25% and 25%.
We started by computing the prices via the Fast Fourier Transform approximation: we defined the Laplace exponent, the characteristic function and the function to be Fourier transformed via three function handles and then we set the parameters of the FFT as follows:

Parameter	value or expression
M	15
N	2^M
dx	0.0025
x_1	$\frac{-dx\cdot(N-1)}{2}$
x	$[x1 : dx : -x1]$
du	$\frac{2\pi}{(dxN)}$
u_1	$\frac{-du\cdot(N-1)}{2}$
u	$[u1 : du : -u1]$

Table 1: Relationships among FFT parameters

where x is the variable representing the moneyness value, u is the variable in the Fourier space and N is the number of intervals. We started by setting $M = 15$ and $dx = 0.0025$ (as reported in the Table1) and we computed the prices of the call options; in a second moment we made also further analysis to understand how the results changed for different values of M (and therefore for different numbers of intervals N) and dx. The integral in the Lewis formula was computed via the function `integralViaFFT`, where we took the real part of the results to neglect the imaginary part which remained because of some approximation errors (indeed we observed that all the imaginary parts were very close to the Matlab's machine epsilon). We finally interpolated the results for values of moneyness x between -25% and 25%.

Afterwards, we computed the call prices via the Quadrature method: we approximated the value of the integral thanks to the Matlab built-in function `integral`, which uses the global adaptive quadrature approximation.

In the end we also computed the call prices via a Monte-Carlo method by doing 10000 simulations of the Gaussian random variable z and of the Inverse Gaussian random variable G (which was simulated by sampling Uniform random variables and then using the inverse cumulative distribution function of the Inverse Gaussian). By using these random variables, we simulated the log-forward prices by applying the Antithetic Variables technique and we computed the prices (and their 95% confidence interval) by discounting the payoffs.

We represented the plot of the call prices computed via the three different techniques in Figure1.

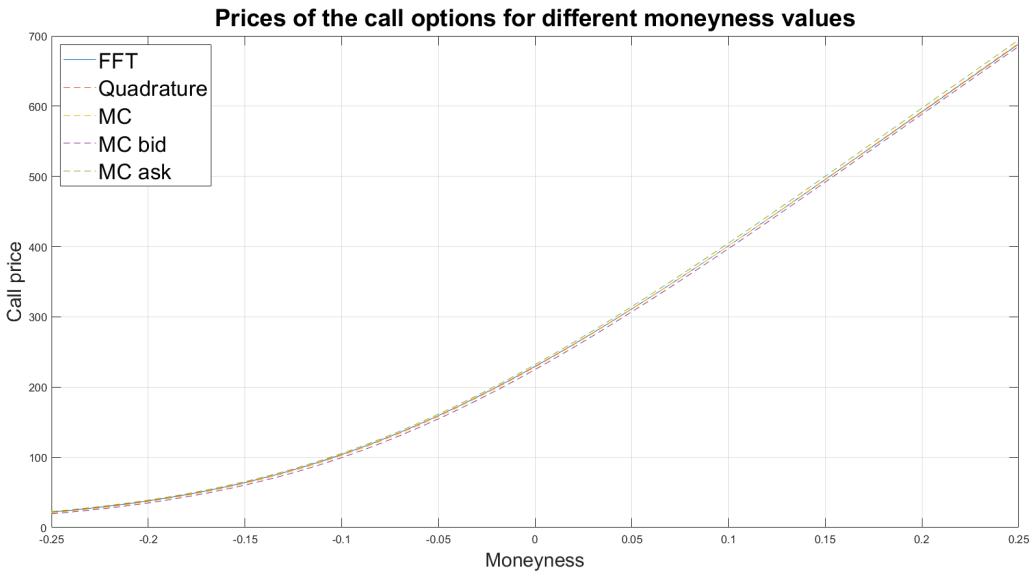


Figure 1: Prices of the call options computed via three different methods plotted for different moneyness values.

We can immediately observe that all the methods give very similar values of the call prices, and from the graph the three results are almost indistinguishable. In all the cases, the more the moneyness increases, the more the call price increases, as we expected, since for higher moneyness values the probability of the stock to be in the money is greater.
To have a quantitative estimation of the accuracy of the methods, we computed the computational times and the L^∞ norm of the difference between the approximated methods (FFT and MC) and the Quadrature method (which we took as the benchmark price) as:

$$|Error|_{L^\infty} = \max_{x=-0.25,\dots,0.25} |Price(x) - Price_{Quadrature}(x)|$$

We reported the results in Table2:

	$\|Error\|_{L^\infty}$	Computational time
FFT	0.0049738567	0.0927205000
MC	0.6597361172	0.1580916000

Table 2: Error with respect to quadrature and computational times of FFT method and MC method

We can notice that even if the Monte-Carlo method still gives a very good approximation, the FFT method error is significantly lower, and his computational time is lower as well. All things considered, for the number of simulations chosen and the parameters' values used, the FFT method seems to be closer to the quadrature one. However, when a software which cannot compute the FFT with build-in functions is used, the Monte-Carlo method is a useful technique to get a very

good approximation of the price.

To have a better understanding of the FFT method, we computed the Call Prices for different values of the number of intervals N and of the step in the moneyness grid dx.

We started by fixing $dx = 0.0025$ and varying the parameter M for values which go from 1 to $M_{max} = 24$ (i.e. we considered $N = 2^M$ intervals each time). We estimated the computational time and the error of the computed prices with respect to the quadrature price for each value of N, and we represented graphically the results in Figure2 and Figure3:

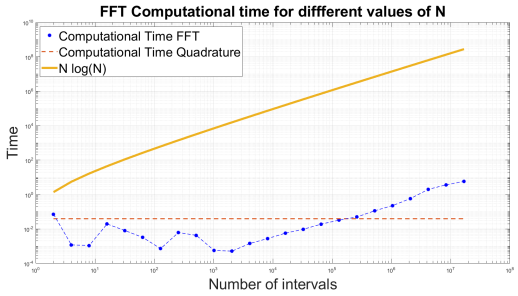


Figure 2: Time to compute the call options' prices via the FFT for different values of N (blue) plotted against the quadrature method computational time (orange) and the function $N \cdot log(N)$ (yellow).

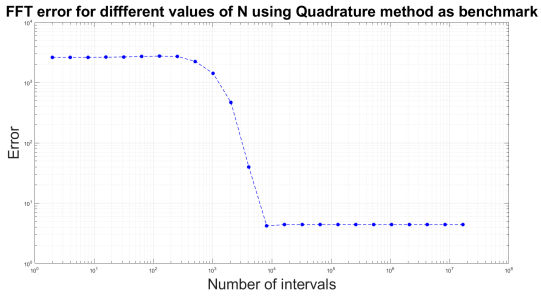


Figure 3: Error of the computed call options' prices via the FFT method for different values of N with respect to the prices obtained via the quadrature method.

For the computational time we start by observing that as N increases, the computational time tends to have the same slope of $N\,log(N)$; in addition, we can see that as long as $N < 2^{19}$, the FFT method is faster than the Quadrature method. From the error's graph we can notice that the error plummets for values of N greater than 2^{12} and then it tends to remain almost constant.
From these analyses, we can conclude that the value $M = 15$ used above gives us a very good trade-off between computational time and precision of the method.

We proceeded by fixing $M = 15$ (since we saw that it is a "good" value) and varying the step of the moneyness grid by multiplying the previously used value $dx = 0.0025$ for the vector $[0.125; 0.25; 0.5; 1; 2; 5; 10]$. The results obtained are graphically represented in the following Figure4 and Figure5:

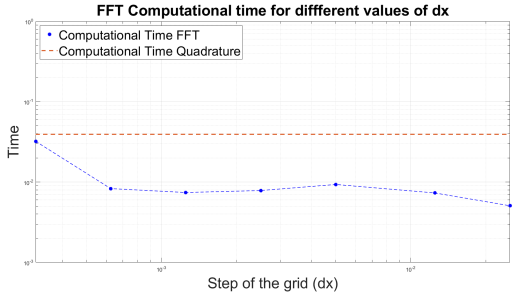


Figure 4: Time to compute call options' prices via FFT for different values of the step dx (blue) plotted against the quadrature method computational time (orange).

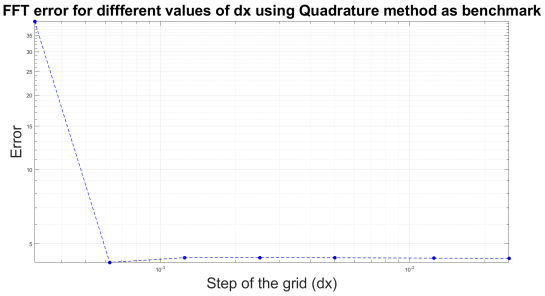


Figure 5: Error of the computed call options' prices via the FFT for different values of the step dx with respect to the prices obtained via the quadrature method.

We can immediately observe that the computational time is always smaller than the computational time for the Quadrature method and it decreases as the grid step becomes greater: we could have expected this result since as the grid becomes less dense, the computation is faster.
For the error we observe a particular behaviour: from a naive point of view it could be said that the smaller the step of the grid, the smaller the error; however in the graph it can be observed that for the smallest step considered $dx = 0.0003125$ the error is significantly bigger than in all the other cases, which all result to have an error of the same order. This can be explained by remembering that in the FFT method there must be a sort of compensation effect between the moneyness step dx and the step in the Fourier space du: there is an inverse proportion between the two variables, described by the relationship

$$dx\, du = \frac{2\,\pi}{N}$$

Therefore, if dx becomes too small, the value of du grows and the computation of the integral via the FFT method becomes less precise.
Moreover, regarding dx, by considering its computational time and its error we can say that the value $dx = 0.0025$ used at the beginning of this exercise is acceptable.

3.2 Facultative case of $\alpha = 2/3$

We concluded this exercise by considering $\alpha = 2/3$, for which we computed the call prices via FFT and quadrature methods. We set the FFT parameters as before ($M = 15$ and $dx = 0.0025$) and we repeated the previously used procedure for both methods, only changing the Laplace exponent, retrieving the formula from the one for a general α in $(0, 1)$.
We plotted the prices obtained via the two methods against the different values of moneyness in Figure6.

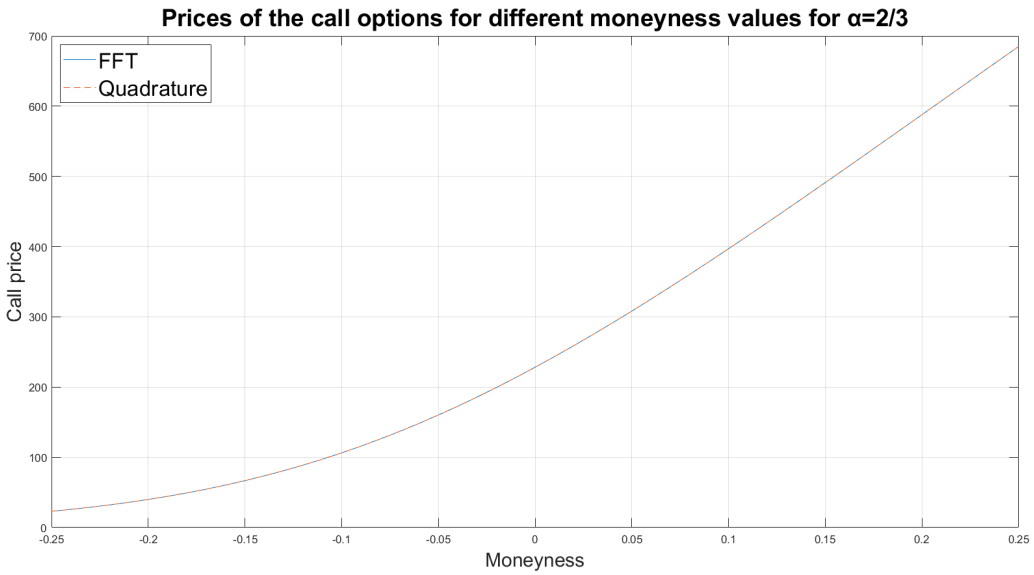


Figure 6: Prices of the call options computed via three different methods plotted for different moneyness values, in the case $\alpha = 2/3$.

From this graph, we can observe that the FFT and the quadrature methods give very close results also in this case; this can be confirmed by computing the L^∞ norm, which results to be

$$|Error|_{L^\infty}^{\alpha=2/3} = 0.0045941766$$

Since we computed the prices of the same call options with two different methods, it would be interesting to understand what are the differences in the obtained results. To analyse this, we started by plotting the obtained prices for different values of alpha via the FFT method on the same plot, obtaining the result shown in Figure7.

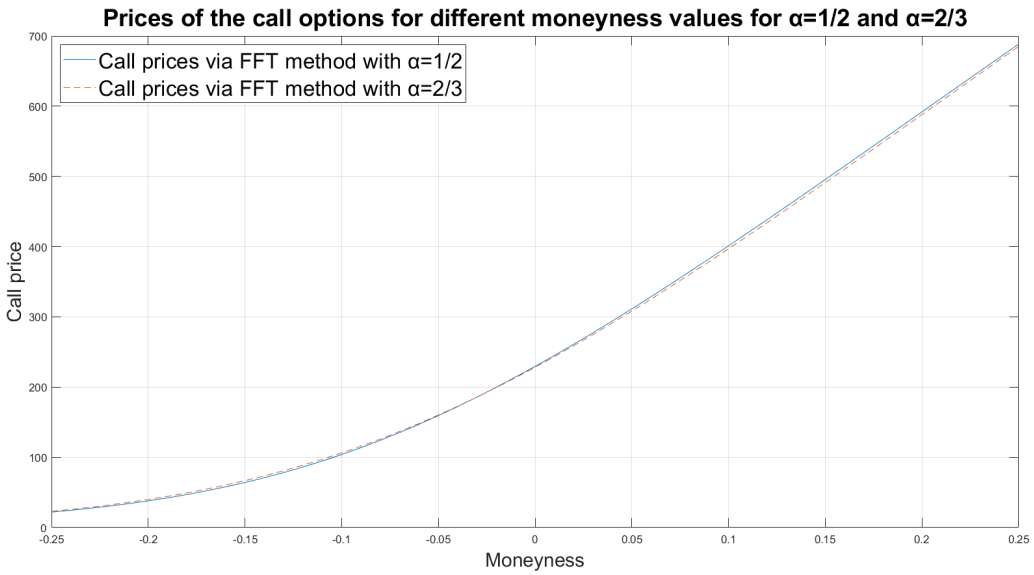


Figure 7: Prices of the call options computed via FFT method, plotted for different moneyness values, in the cases $\alpha = 1/2$ and $\alpha = 2/3$.

From this graph we can observe that the two models give very close values, even if the two lines do not exactly overlap: the prices computed via the model with $\alpha = 1/2$ seem to be greater than the ones computed with $\alpha = 2/3$ for bigger moneyness values, while the opposite trend is observed for lower moneyness values. This is well graphically represented in Figure8, where we plotted the difference between the prices, computed as $Differenence = Prices_{\alpha=1/2} - Prices_{\alpha=2/3}$

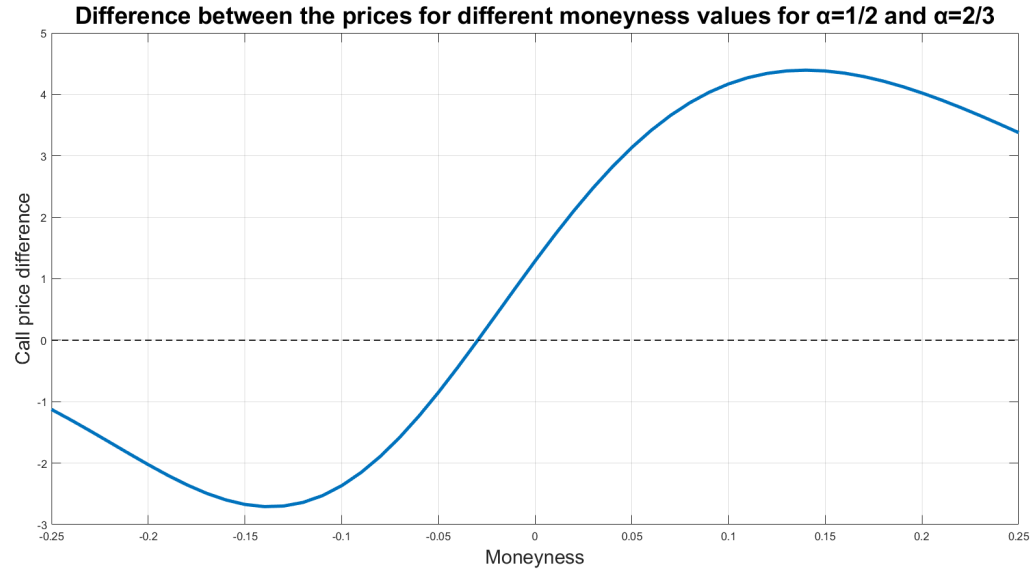


Figure 8: Prices of the call options computed via FFT method, plotted for different moneyness values, in the cases $\alpha = 1/2$ and $\alpha = 2/3$.

In general, the two methods give different prices for out of the money or in the money call options, however this differences are never severly significant, since the greatest difference in percentage for the call prices is equal to 5.42%.

We concluded our analysis by asking ourselves whether if significant differences in terms of precision of results were present in the two methods with different values of α. We computed the error of the call prices with respect to the quadrature price for different values of N in the case $\alpha = 2/3$ and we plotted the obtained results together with the errors computed for $\alpha = 1/2$, as shown in Figure9.

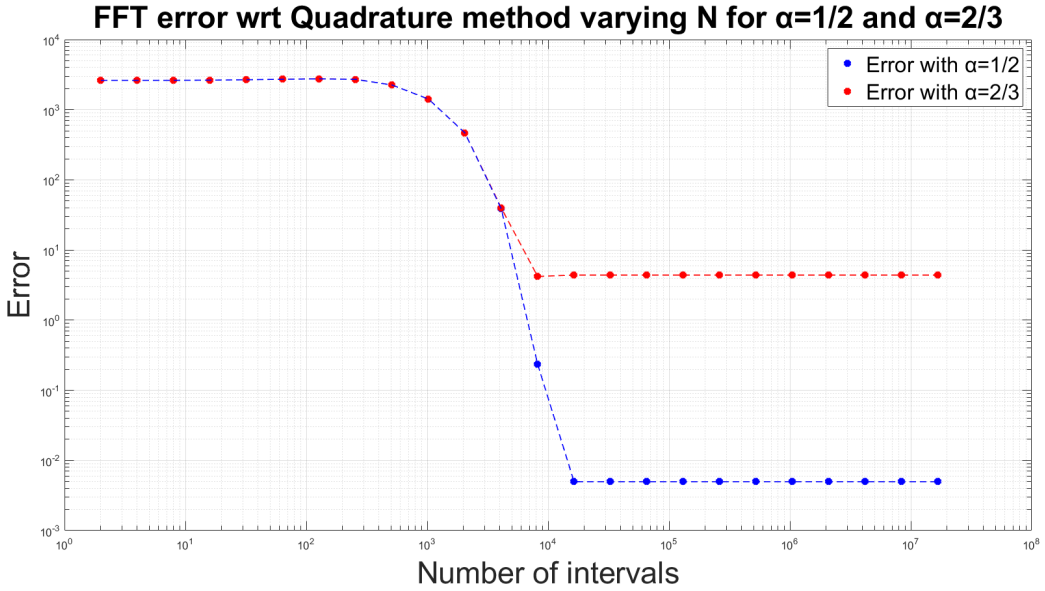


Figure 9: Error of the computed call options' prices via the FFT for different values of N with respect to the prices obtained via the quadrature method for $\alpha = 1/2$ (blue line) and $\alpha = 2/3$ (red line).

From these results we can infer that the model with $\alpha = 1/2$ reaches the errors' limit for a lower number of intervals; however, the error with respect to the quadrature method of the model with $\alpha = 1/2$ reaches a significant lower limit value.

4 Volatility surface calibration

We have been tasked with calibrating the parameters of a Normal Mean Variance Mixture model using the quoted volatilities from Question 2, with a specified value of $\alpha = 1/3$. Initially, we obtained market prices using the Black formula and computed model prices using the FFT method. Next, we set up the algorithm parameters, considering the precision observed in previous steps. Then, we defined a pricing function dependent on the three model parameters:

- σ (average volatility)

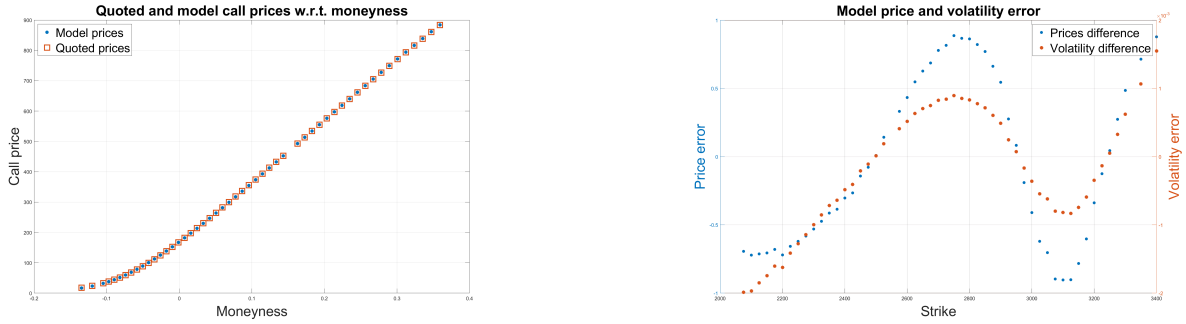
- η (volatility skew)
- κ (vol-of-vol)

We calibrated the model using MATLAB's `lsqnonlin` function. As input, we utilized the function representing the discrepancy between quoted and model prices with respect to the mentioned parameters. The minimization process in `lsqnonlin` corresponds to optimizing the L^2 metric, aiming to minimize the squared differences between quoted and model prices. The calibrated parameters are as follows:

Parameter	**Calibrated Value**
σ	0.124191
η	7.044890
κ	1.672531

Table 3: Calibrated Parameters

The σ value appears reasonable as it signifies the average volatility across various moneyness levels. Notably, the skew parameter η exhibits a high magnitude, indicating pronounced asymmetry in the quoted smile, consistent with the typical positive skew observed in equity derivatives. The mean squared error (MSE) of prices in the calibration process is 0.362838. Examination of the graphs reveals no discernible or explanatory patterns in price discrepancies. The associated volatility MSE is 0.000121, showing a positive correlation between the signed price differences and volatilities. This correlation aligns with the theoretical expectation of a positive relationship between prices and volatilities. Calibrating the model directly with price data rather than volatilities results in the relative error in volatilities being typically around one order of magnitude higher than that of prices.



(a) Market prices vs. model prices

(b) Differences between quoted and model values

Figure 10: The left graph shows the market prices and the model's ones using calibration output parameters to price the call. The right plot highlights the signed differences between quoted and model values for prices and volatilities.

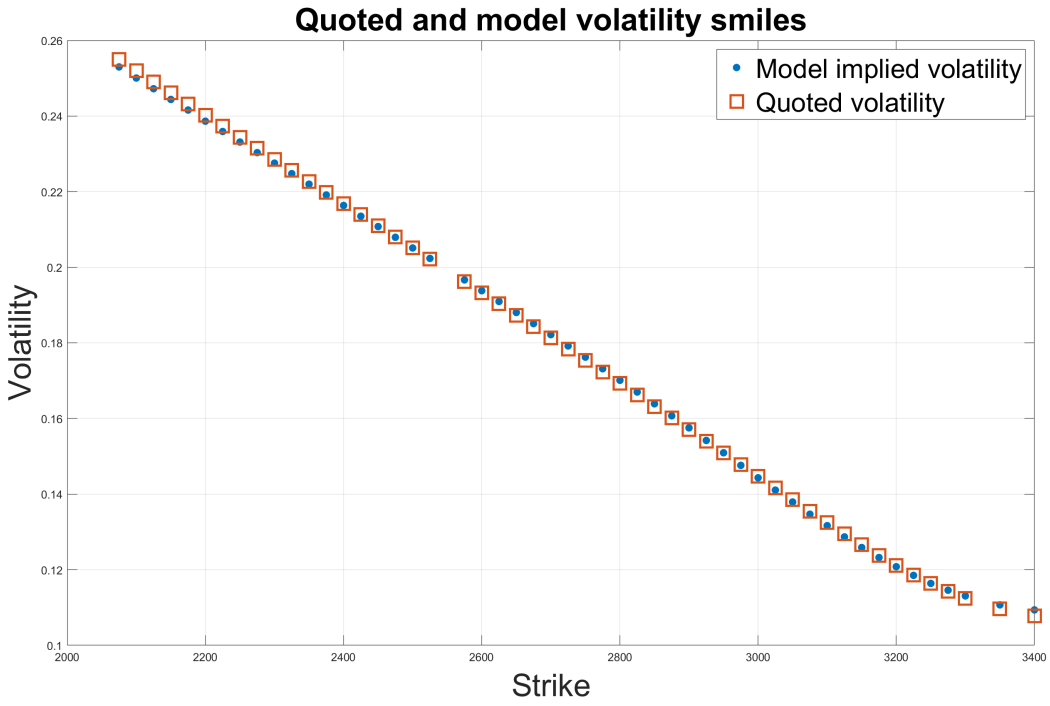


Figure 11: Volatility smiles: blue dots represent the market implied volatilities, the red squares are the model's ones .