Optimization In Finance: Project Report

I A Delta-neutral Trading Strategy Based on Implied Volatility Surface

II Calibration of Heston Model with An Application of FFT And Genetic Algorithm

Jiehao Liu

Director: Professor Reza Kamaly

Rackham Graduate School, University of Michigan

Ann Arbor, USA

August 3, 2014

Abstract

Trading and Calibration are two very essential daily tasks for a quant. In this project, we build a delta-neutral strategy for trading equity option based on implied volatility surface. We build our position at the beginning of the trading hour by comparing the market price and the model price which given by applying previous day's implied volatility. A threshold is set for this price discrepancy, and a parameter optimization is carried out to determine how large it should be to make the most profitable strategy. We hedge our position simply by using delta and liquidate our position at the end of the trading hour. Our data shows that this is not a very profitable trading strategy, it cannot cover the bid-ask spread 'fee' we pay for each trade, basically the more we trade the more we loose. We use a reverse trading strategy (by taking opposite position) to testify our judgment about why we loose.

Heston model are implemented in three ways in this project: a FFT approach and other two numerical integral approaches. Our result shows FFT approach does give us time efficiency but lack of accuracy in pricing. To perform the calibration of Heston model, a local optimization method and a global optimization method-genetic algorithm are used. Genetic algorithm is very time consuming but could give us a more reasonable value which could be used as initial point if we have to carry out calibration forward rolling. Our calibration result shows that for options with short term the standard error for parameters are larger, but sum of square error is smaller. From a moneyness perspective, the Heston model works very well for options out the money.

Acknowledgements

I would like to thank all the people who have generously given me kind help and instructions. Also, I feel grateful to the authors whose intellectual work have been referred to in this project.

For all the typing errors and code bug, Jiehao Liu will assume responsibility for them. Matlab code for each chapter are placed in the code file, please run them correspondingly with the raw data. If you find trouble running the code please contact jiehao@umich.edu.

Contents

List of Figures

Chapter 1

Black-Scholes Model and Formula

1.1 Derivation of BS Formula

Assume we want to price one T-contingent claim on Stock which price at T is denote as $S(T)$ and the contingent claim's payoff is

$$\Phi(S(T))$$

. Start with the Black-Scholes Model:

$$dB(t) = rB(t)dt, \tag{1.1}$$

$$dS(t) = \alpha S(t)dt + \sigma S(t)dW(t) \tag{1.2}$$

Then, lets construct a self-finance portfolio which contain stock and bond and use it to replicate the contingent claim:

$$\begin{aligned} dV(t) &= r(V(t) - h^* S(t))dt + h^* dS(t) && (1.3) \\ &= rV(t)dt - rh^* S(t)dt + h^* \alpha S(t)dt + h^* \sigma S dW(t) && (1.4) \\ V(T) &= \Phi(S(T)) && (1.5) \end{aligned}$$

Meanwhile, we assume the T-contingent claim's price is a continuous function of $S(t)$, denote as $F(t, S(t))$. We assume it's in this form because if it could be replicated by a portfolio consists of stock and bond then its value should be the same as the portfolio which is of course in the form of $F(t, S(t))$, by it$\widehat{o}$ formula we get the price dynamic as:

$$dF(t, S(t)) = \left\{ F_t + \alpha S F_s + \frac{1}{2}\sigma^2 S^2 F_{ss} \right\} dt + \sigma S F_s dW(t), \tag{1.6}$$

To make the self-financing portfolio a replicating of the contingent claim, we need

$$dV(t) = dF(t, S(t)) \tag{1.7}$$

$$V(T) = F(T, S(T)) \tag{1.8}$$

To set $h^* = F_s(t, S(t))$, which denote the delta of the option (or number of stock we need to hold to do the replicating). Then plug (2.4) and (2.6) into (2.7), we will get the wanted Black-Scholes partial differential equation:

$$F_t + rSF_s + \frac{1}{2}\sigma^2 S^2 F_{ss} = rF \tag{1.9}$$

Meanwhile, plug (2.5) into (2.8), we will get boundary condition of this equation

$$F(T,S(T)) = \Phi(S(T)) \tag{1.10}$$

Now, if we let

$$\Phi(S(T)) = \mathbf{max}\{S(T)-K,0\}$$

, which is the payoff of a call option with strike price K, then we can solve this PDE and get the famous Black-Scholes formula for a call option:

$$F(t,s) = SN[d_1(t,s)] = e^{-r(T-t)}KN[d_2(t,s)] \tag{1.11}$$

where,

$$d_1(t,s) = \frac{ln(\frac{S}{K}) + (r+\frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}} \tag{1.12}$$

$$d_2(t,s) = d_1(t,s) - \sigma\sqrt{T-t} \tag{1.13}$$

1.2 Critiques about BS Model

Black-Scholes model is widely used in both the industry and academia, however, that's not saying that the price given Black-Scholes Model is the 'right' price to pay for an option. The greatness of Black-Scholes is that it specifies a stochastic process for stock price movement and then use an dynamic delta-hedging (or replicating) strategy to give a no arbitrage price of the option. The dynamic hedging is very essential for this price to be the 'right' price. If we can really do a perfect hedging, then this price should be the 'right' price, at least it should not diverge form the market price too much. As we described before, the hedging is based on the assumption that the stock price movement follows Geometric Brownian Motion.

$$dS_t = \alpha S_t dt + \sigma S_t dW(t)$$

For this to be true, that is to say the return should be normally distributed and the volatility would be a constant number, plus, the price should be continuous which means there should be no jumps.

For a moment, let's assume that other assumptions such as no trading cost and market is complete do hold. Even though, we could still see that the former three assumption do not hold in reality. To adjust these issues, researchers have developed other models to describe the movement of the underlying asset's price. Now, price movement is modeled with jumps and stochastic volatility to make it more realistic. In the second part we'll introduce Heston model, a model with stochastic volatility, and we'll do a calibration of the mode. But in the first part, let us bear with the imperfection of Black-Scholes model and use it to get the so called implied volatility surface.

Chapter 2

Delta-Neutral Trading Strategy

In this part, a delta-neutral trading strategy is built by using the implied volatility surface. We assume the price given by the implied volatility from previous day's last trading record is the "correct" price to pay for the option. If there is a discrepancy between this price and the market price, we'll take advantage of it by implementing a delta-neutral trading strategy. The reason we hedge it against stock price movement is because we only want to profit the price discrepancy of the option and don't want to get affected by the movement direction of the stock market. To run our strategy in the laptop, from raw data to the most desired result for the 14 trading days, it will take about 2 minutes, including dealing with the raw data and strategy optimization.

2.1 Dealing with the Data

We have a data base of 14 consecutive trading days, which is not organized in a way to easily carry out the trading strategy. For each day, we need to find out 4 sets of data that concerns us.

. **dayMornOpt** This set consists of all the data needed to find BS price of an option. They're the first trading record of each option.

. **dayEndOpt** This set consists of all the data needed to find the implied volatility of an option.They're the last trading record of each option.

. **datBidAskM** This set consists of all the data needed when we building our arbitrage strategy. They're the first bid-ask record of the stock and the option.

. **datBidAskE** This set consists of all the data needed when we liquidating our positions. They're the last bid-ask record of the stock and the option.

This work is done by filtering and matching,

1. **Filtering** This is for spotting the data that concerns us as described in the above.

2. **Matching** This is for associating implied volatility from previous day for for each option to the current trading day price data.

Once we get these related data, life would be way easier.

2.2 'NaN' Return Issue with `blsimpv`

One problem we need to deal is that the Matlab built in function `blsimpv` could return 'Na' sometimes. This is because the built in algorithm could not find a root to the equation in a required domain. For our trading implementation, we don't actually need the accurate implied volatility, all we need is to use implied to get the model price of an option and compare it with the market price. The compare result is key to us, so, for those option which return a 'NaN' value when calling `blsimpv` , we will simply replace it with a very small but positive value 0.001, and this will give us the right compare result.

2.3 A Brief Description of Our Trading Strategy

Our trading strategy is very simple and straightforward.

Step One Calculate the implied volatility by using the last record for each option form previous day.

Step Two Calculate the BS price of the option by using the implied volatility get from previous day and applying it to the first trading record of current day.

Step Three Compare the market price and the model price, short the ones overvalued and long the ones undervalued.A threshold would be set here. Meanwhile, build a corresponding hedging position.

Step Four Liquidate our position by using the last trading record.

Some interesting issues here are:

1. **Hedging** We need to do a hedging on one-by-one basis, since we enter position for each position in different time point. This will increase trading cost in real life scenario, but here we don't need to consider associated trading cost. However, it's still worth notice that we can do hedging on a portfolio level in real situation to lower the cost.

2. **Predictable** A trading strategy must be predictable, that is to say, we must trade only on the information we already have, we can't looking into the future. For our liquidating rule, this looks like be violated, since we can't tell in real situation whether this is the last trading or not. However, our rule don't have to be this, we can't liquidate on any time point and we're actually not using any forward information about the market price. We set this rule only because it's straightforward.

Then this leave us with one major problem, optimizing our trading strategy to choose the best parameter, that is to say to find the best threshold.

2.4 Optimization and Result

We trade based on the discrepancy between the market price and the model price. A threshold is set here to determine how large the difference should be for us to build the most profitable strategy.Denote the threshold as H, then for the overvalued option

$$\frac{\textbf{Marketbid-Modelprice}}{\textbf{Modelprice}} > H$$

For the undervalued option

$$\frac{\textbf{Modelprice-Marketask}}{\textbf{Modelprice}} > H$$

Here, for the value of H, we start from $\frac{1}{1000}$ to $\frac{200}{1000}$.

Two results are listed below, one is for trading 13 days, that means we trade every day from day2. We can't trade on day1 since we need previous volatility data. The other result is trade on every day except the second last one, since we found data from that day is abnormal, and result is significant different when we leave out the second last day.

2.4.1 Result one: Trading on Every 13 Day

Let's look at how's our cumulative gains when we choose different H. Figure 2.1 shows that basically, the smaller H is, the more we loose. This make sense, since for small H means we gonna enter more trade with very low chance of making a profit, however we're suffering from 'paying' the bid-ask spread. When H is very larger, we bare enter any trade, so we don't loose or make a profit. So the figure converge to zero.

Fortunately, we do make some profit when the right H is choosen. From our result,

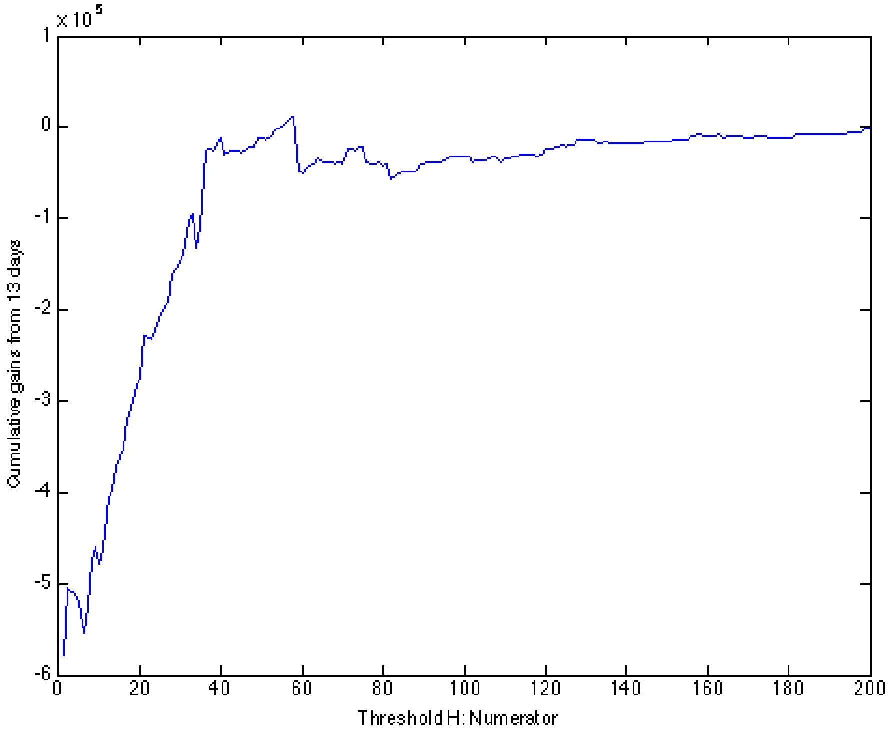

Figure 2.1: Cumulative gains when different H is set.

the best H is $\frac{55}{1000}$. Figure 2.2 shows what the profit looks like for each trading day when $H = \frac{55}{1000}$. One might notice that this is not very attractive, since make a big profit in the second last day, excepting that we barely make a stable profit. So we might want have a look at how things would be if we leave out the second last trading day.

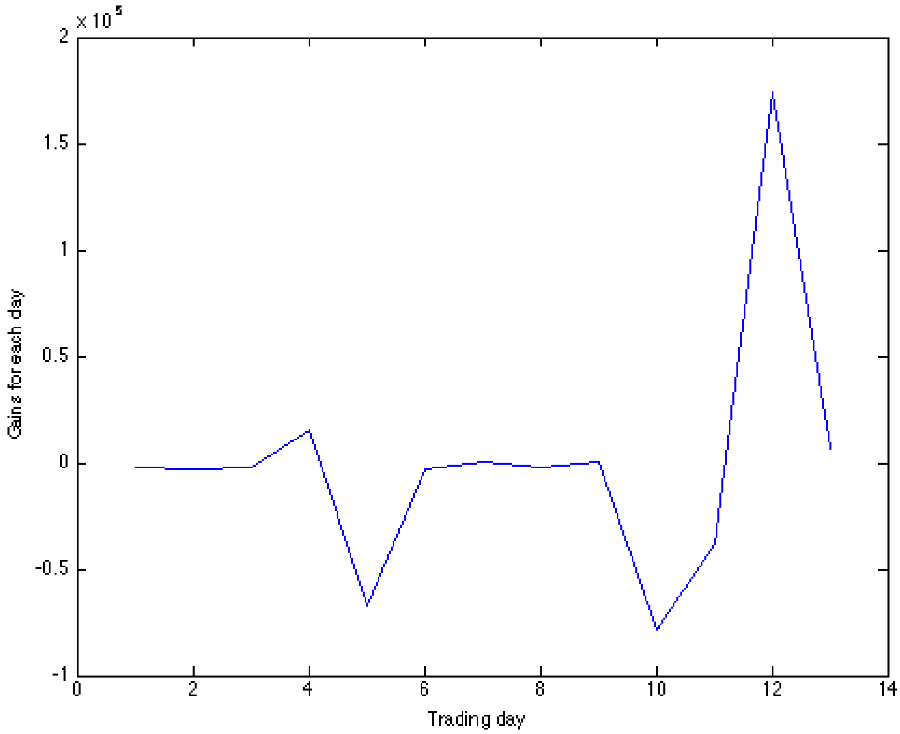

Figure 2.2: Gains for each trading day when $H = \frac{55}{1000}$.

2.4.2 Result two: Leave Out the Second Last One

As we expected, Figure 2.3 shows when we leave out the second last trading day. We can hardly make a profit no matter what value we set for H. Meanwhile, one might find that we loose on a very steady basis. So, what exactly happened here to make us loose in such a steady way? Can we reverse the trading strategy and make a profit?

2.5 Why We Loose: Reverse the Strategy

As we predict before, we loose is probably because the bid-ask spread we paid for each trade. Is this true? How we could testify our guess? Here, we build a reverse trading strategy just by take the opposite position as the one we build in previous section. If we loose again, that is to say, we loose simply for suffering from the bid-ask spread. And the strategy by trading on implied volatility is not a profitable one.
Figure 2.4 shows, as we expected, we loose again and the profit chart looks just like the one from the previous strategy. The common factor here is the bid-ask spread,

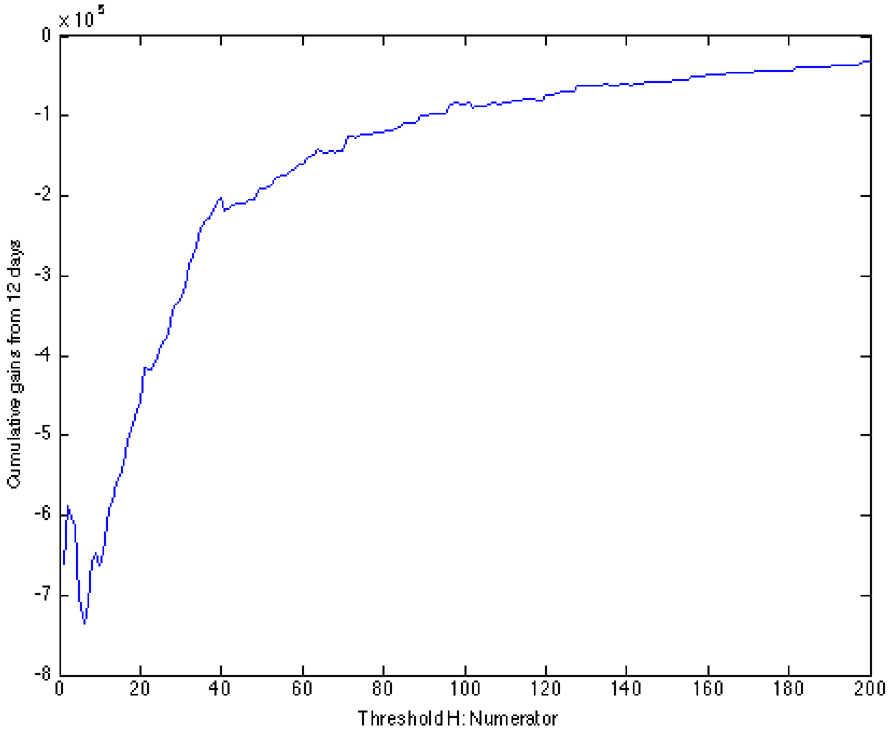

Figure 2.3: Cumulative gains when different H is set.[Second last day is out]

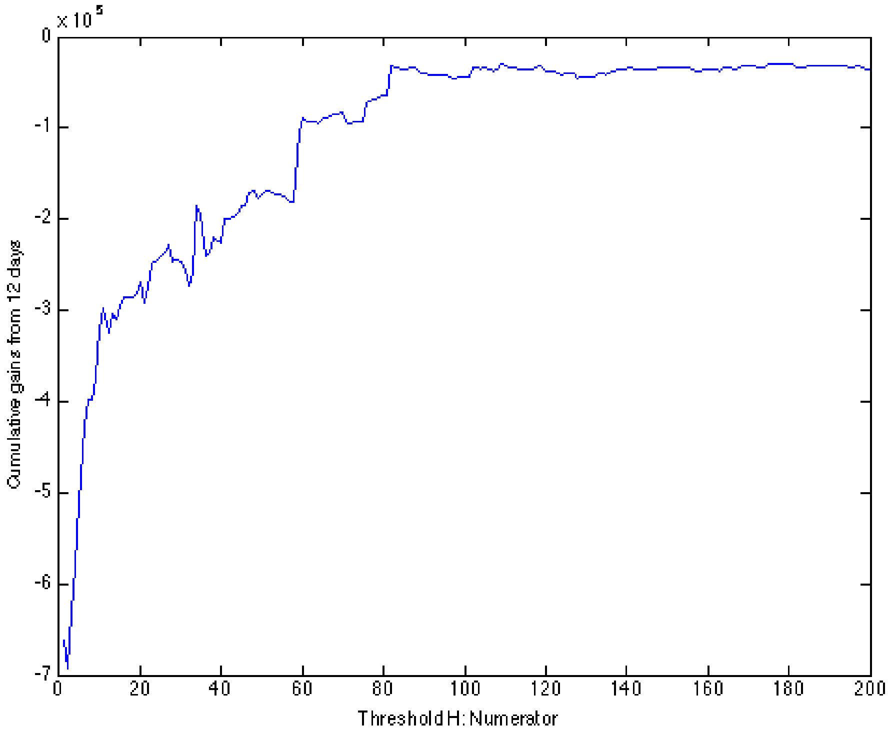

Figure 2.4: Reverse strategy: Cumulative gains when different H is set.[Second last day is out]

so, we can conclude that we loose because of bid-ask spread we pay for each trade.

2.6 Conclusion

If we leave out the second last trading day, the trading strategy based on implied volatility is not very profitable. At least, it can't cover the bid-ask spread 'fees' we pay for each trade. We haven't done a thorough investigation about what happened in the last second trading day, even if everything is right, this strategy is not a very good candidate for trading, it's too volatile.

Chapter 3

Heston Model

In this chapter, Heston Model will be introduced. In the first section, a replicating approach is used to derive the valuation equation of Heston Model. Then, characteristic function is used to solve this valuation equation and would give a semi-closed formula. The last section would deal with the interpretation of parameters in the Heston Model.

3.1 The Valuation Equation

Suppose that the stock price S and its variance v satisfy the following SDEs:

$$dS_t = u_t S_t dt + \sqrt{v_t} S_t dW_t, \tag{3.1}$$
$$dv_t = \alpha(S_t, v_t, t)dt + \sigma\beta(S_t, v_t, t)\sqrt{v_t} dZ_t, \tag{3.2}$$

where $dW_t dZ_t = \rho d_t$.

Apparently, this market is not complete since there is only one risky asset and two source of randomness. To price a contingent claim on the risky asset, we need to assume there is another traded contingent claim in the market.

Let U and U_1 denote the price of the contingent claim we want to price and the already traded contingent claim respectively. Also, let Δ and Δ_1 indicate the position we hold on the underlying stock S and the U_1 correspondingly.

Build a risk-less portfolio:

$$\Pi = U - \Delta S - \Delta_1 U_1.$$

The change in this portfolio in a time dt is given by

$$\begin{aligned} d\Pi = &\left\{ \frac{\partial U}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U}{\partial S^2} + \rho\sigma v\beta S \frac{\partial^2 U}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U}{\partial v^2} \right\} dt \\ &- \Delta_1 \left\{ \frac{\partial U_1}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U_1}{\partial S^2} + \rho\sigma v\beta S \frac{\partial^2 U_1}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U_1}{\partial v^2} \right\} dt \\ &+ \left\{ \frac{\partial U}{\partial S} - \Delta_1 \frac{\partial U_1}{\partial S} - \Delta \right\} dS \\ &+ \left\{ \frac{\partial U}{\partial v} - \Delta_1 \frac{\partial U_1}{\partial v} \right\} dv \end{aligned}$$

To make the portfolio instantaneous risk-free,choose

$$\frac{\partial U}{\partial S} - \Delta_1 \frac{\partial U_1}{\partial S} - \Delta = 0$$

to eliminate dS terms, and

$$\frac{\partial U}{\partial v} - \Delta_1 \frac{\partial U_1}{\partial v} = 0$$

to eliminate dv terms.This leaves

$$\begin{aligned} d\Pi = & \left\{ \frac{\partial U}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U}{\partial S^2} + \rho\sigma v \beta S \frac{\partial^2 U}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U}{\partial v^2} \right\} dt \\ & - \Delta_1 \left\{ \frac{\partial U_1}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U_1}{\partial S^2} + \rho\sigma v \beta S \frac{\partial^2 U_1}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U_1}{\partial v^2} \right\} dt \\ & = r\Pi dt \\ & = r(U - \Delta S - \Delta_1 U_1) dt \end{aligned}$$

Here, the arbitrage argument is used.Since the portfolio is risk-free, the return must be risk-free rate. Rearrange the equation, collecting all the U on the left-hand side and all the U_1 on the right-hand side, gets

$$\begin{aligned} & \frac{\frac{\partial U}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U}{\partial S^2} + \rho\sigma v \beta S \frac{\partial^2 U}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U}{\partial v^2} + rS\frac{\partial U}{\partial S} - rU}{\frac{\partial U}{\partial v}} \\ = & \frac{\frac{\partial U_1}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U_1}{\partial S^2} + \rho\sigma v \beta S \frac{\partial^2 U_1}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U_1}{\partial v^2} + rS\frac{\partial U_1}{\partial S} - rU_1}{\frac{\partial U_1}{\partial v}} \end{aligned}$$

The left-hand side is a function of U only and the right-hand side is a function of $U1$ only. The only way that this can be is for both sides to be equal to some function f of the independent variables S, v and t. Deduce that

$$\frac{\partial U}{\partial t} + \frac{1}{2} v S^2 \frac{\partial^2 U}{\partial S^2} + \rho\sigma v \beta S \frac{\partial^2 U}{\partial v \partial S} + \frac{1}{2}\sigma^2 v \beta^2 \frac{\partial^2 U}{\partial v^2} + rS\frac{\partial U}{\partial S} - rU = -(\alpha - \phi\beta\sqrt{v})\frac{\partial U}{\partial v} \quad (3.3)$$

where, without loss of generality, the arbitrary function f of S, v and t has been written as$(\alpha - \phi\beta\sqrt{v}$,where α and β are the drift and volatility functions from the SDE (5.2) for instantaneous variance.

Further, the Heston Model choose $\alpha(S, v_t, t) = k(\theta - v_t)$ and $\beta(S, v, t) = 1$ in equation (5.3). Thus, give us the stock price process:

$$dS_t = (r)S_t d_t + \sqrt{v_t} S_t dW_t \quad (3.4)$$

,and process for the stochastic variance

$$dv_t = k(\theta - v_t)d_t + \sigma\sqrt{v_t} dZ_t \quad (3.5)$$

where,

1. k: Mean reversion rate
2. θ: Long run variance
3. σ: Volatility of variance

3.2 The Semi-Closed Solution

Suppose the contingent claim is a European call option with Strike price K and expirate time T. Now, we'll give the semi-close solution to the above equations without listing the derivation. The solution is in the form as:

$$C(S_t, v_t, t, T) = S_t P_1 - K \exp^{-r(T-t)} P_2 \tag{3.6}$$

where the first term is the present value of the underlying (spot) asset, and the second term is the present value of the strike price payment. And P_1, P_2 are given as

$$P_j(x, v_t, T; ln[K]) = \frac{1}{2} + \frac{1}{\pi} \int_0^\infty Re\left[\frac{\exp^{-iuln[K]} f_j(x, v_t, T; u)}{iu}\right] du \tag{3.7}$$

where, $x = ln(S_t)$, f_1 and f_2 are given as

$$f_j(x, v_t, T, u) = exp\{C(\tau, u) + D(\tau, u)v_t + iux\} \tag{3.8}$$

$$C(\tau, u)) = rui\tau + \frac{a}{\sigma^2}\left[(b_j - \rho\sigma ui + d)(\tau) - 2ln\left(\frac{1 - ge^{d\tau}}{1 - g}\right)\right] \tag{3.9}$$

$$D(\tau, u) = \frac{b_j - \rho\sigma ui + d}{\sigma^2}\left(\frac{1 - e^{d\tau}}{1 - ge^{d\tau}}\right) \tag{3.10}$$

$$g = \frac{b_j - \rho\sigma ui + d}{b_j - \rho\sigma ui - d} \tag{3.11}$$

$$d = \sqrt{(\rho\sigma ui - b_j)^2 - \sigma^2(2u_j ui - u^2)} \tag{3.12}$$

for $j = 1, 2$, where,

$$u_1 = \frac{1}{2}, u_2 = -\frac{1}{2}, a = k\theta, b_1 = k + \lambda - \rho\sigma, b_2 = k + \lambda$$

3.3 Fit the Volatility Surface

Empirical studies have shown that an asset's log-return distribution is non- Gaussian. It is characterized by heavy tails and high peaks. There is also empirical evidence and economic arguments that suggest that equity returns and implied volatility are negatively correlated. This departure from normality plagues the Black-Scholes model with many problems. In contrast, Heston's model can imply a number of different distributions by choosing different values for the parameters.And doing this, it can be used to fit different volatility surface in the market.

As an illustration, here list three sets of parameters which will give us volatility surfaces in different shape.Assuming that there is an underlying asset whose current value is 1 and. The strike price of the options traded varies from 0.8 to 1.2 and they have maturities from 0.5 to 3 years.

Heston Model would be to calculate the associated prices of these options, and then treat these prices as market prices and calculate the implied volatility, since we have three sets of different parameters, we'll get three volatility surface. Of course, in real situation, logic does not work in this way and this is only for our illustration

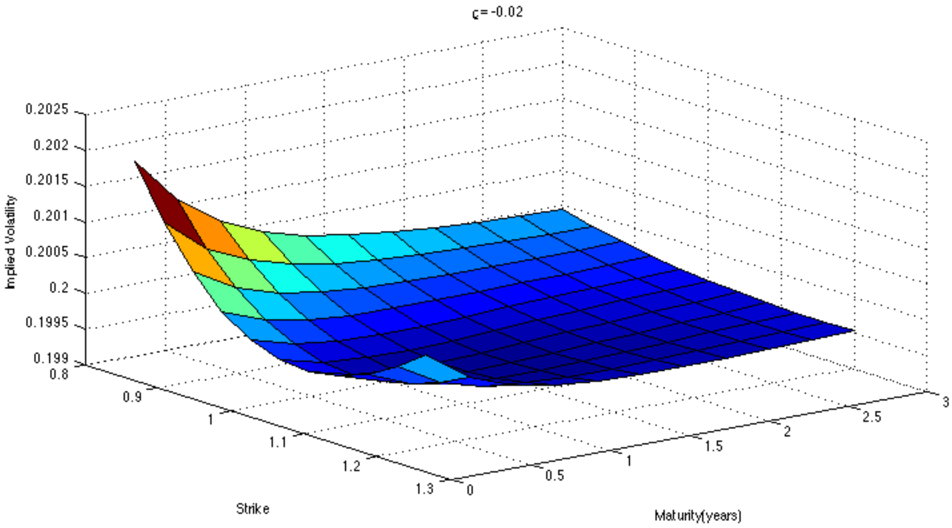

Figure 3.1: Implied volatility surface: $\rho = -0.02, k = 2, \theta = 0.04, \sigma = 0.1, v_0 = 0.04, r = 1\%, S_0 = 1$, strikes: $0.8 - 1.2$, maturities: $0.5 - 3$ years

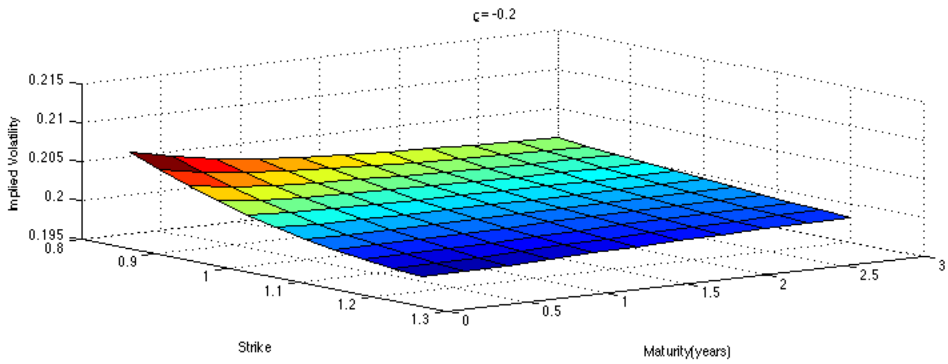

Figure 3.2: Implied volatility surface: $\rho = -0.2, k = 2, \theta = 0.04, \sigma = 0.1, v_0 = 0.04, r = 1\%, S_0 = 1$, strikes: $0.8 - 1.2$, maturities: $0.5 - 3$ years

purpose. In real situation, we will get the market price from the market and use it to calibration the Heston Model, which we will handle later.

Figure 3.1 has $\rho = -0.02$, which means the volatility of return is almost not related to the value of return. It's not surprising to see that this gives a volatility surface which is very similar to what could be got from FX option market. One can even say the volatility smile on the edge.

Figure 3.2 has $\rho = -0.2$, which means the volatility of return is slightly negative related to the value of return. This situation is just like the stock market and without any surprise the volatility surface do look like the volatility surface in the stock market which shows a volatility smirk on the edge.

Figure 3.3 has $\rho = 0.2$, volatility with this shape is not easy to see in the stock market, however, in the commodities market, it do exist due to disequilibrium of supply and demand.

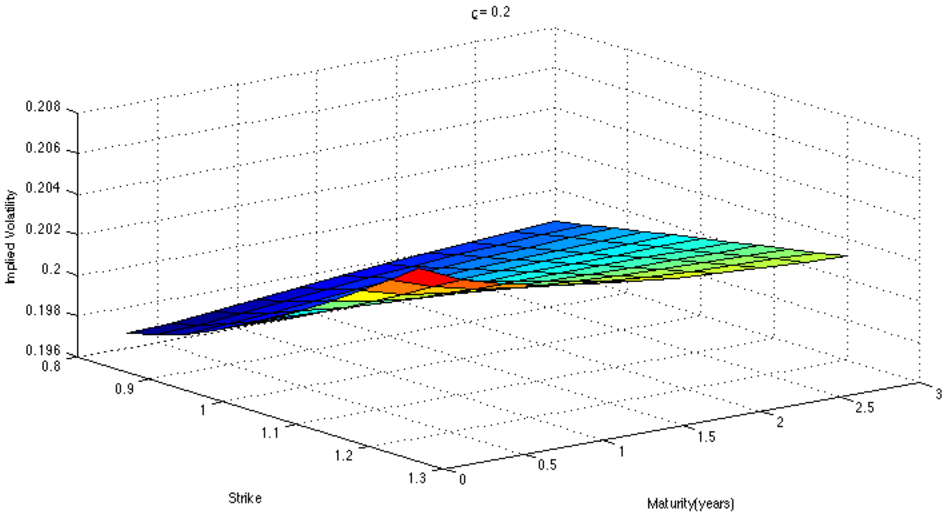

Figure 3.3: Implied volatility surface: $\rho = 0.2, k = 2, \theta = 0.04, \sigma = 0.1, v_0 = 0.04, r = 1\%, S_0 = 1$, strikes: $0.8 - 1.2$, maturities: $0.5 - 3$ years

Only ρ is adjusted here, one can get more volatility surface with various shape by changing the value of more parameters. The above illustration shows how powerful Heston Model could be when fitting the volatility surface. And that's one the main reasons why it gain popularity in the industry.

Chapter 4

Numerical Implementation of Heston Model

The semi-closed solution (5.6)-(5.12) seem a little bit daunting at first glance, however, it's quite straightforward when implemented in Matlab. The major issue here comes from (5.8), we need do a numerical approximation for the integral. In the rest of this chapter, three methods would be employed and discussed. A self-written numerical integral, a built-in numerical integral from Matlab and FFT. We need do some preparation work before we can utilize the speed advantage of FFT.

4.1 Numerical Integral Approach

Again, Let's look at (5.7)

$$P_j(x, v_t, T; ln[K]) = \frac{1}{2} + \frac{1}{\pi}\int_0^{\infty} Re\left[\frac{\exp^{-iuln[K]} f_j(x, v_t, T; u)}{iu}\right] du$$

Which is an improper integral. Its upper limit is infinite. Of course, when doing numerical integral, one has to choose a finite number for this upper limit. The question remain to ask is how large should we choose. No doubt, a larger number would give us more accuracy, but,in the mean while, more time consuming. Fortunately,a quick plot of the integrand will reveal that it converges very quickly to zero. In this case, an upper limit of 100 would sufficient our accuracy demand.

Matlab's built in function `q = quadl(fun,a,b)` approximates the integral of function fun from a to b, to within an error of 10^6 using recursive adaptive Lobatto quadrature. This provides a very accurate way to do the numerical integral. Also, a naive way to do numerical integration is introduced here as a comparison, in which the integral is treated just as a Riemann sum.

4.2 Fast Fourier Transform Approach

To make use of the speed advantage of FFT, we need to rederive a solution to the model, which would make a discrete approximation possible. FFT cannot be used to evaluate the integral (5.7), since the integrand is singular at the required evaluation point $u = 0$. In the next section, an alternative solution is provided and then a discrete approximation is given which will allow us to apply FFT in calculation.

4.2.1 Rederive formula for Call Option

In order to apply FFT, we need to get a pricing formula which could be approximated by something like discretized Fourier transform. The formula by Heston couldn't satisfy this. In this section, we develop analytic expressions for the Fourier transform of an option price. Fourier transforms are expressed in terms of the characteristic function of the log price.

Let $x_t := lnS_t$ and $k := lnK$, where K is the strike price of the option. Then the value of a European call, with maturity T, as a function of k is,

$$C_T(k) = e^{-rT} \int_k^\infty (e_T^x - e^k) f_T(x_T) dx_T \tag{4.1}$$

where $f_T(x)$ is the risk neutral density function of x, dened above. Carr and Madan dene a modied call price function,

$$c_T(k) = e^\alpha k C_T(k), \alpha > 0 \tag{4.2}$$

α is referred to as the dampening factor.The Fourier transform and inverse Fourier transform of $C_T(k)$ is,

$$F_{c_T}(\phi) = \int_{-\infty}^\infty c_T(k) dk \tag{4.3}$$

$$c_T(k) = \int_{-\infty}^\infty F_{c_T}(\phi) d\phi \tag{4.4}$$

Substituting (6.4) into (6.2),

$$\begin{aligned} C_T(k) &= e^{-\alpha k} c_T(k) && (4.5) \\ &= e^{-\alpha k} \frac{1}{2\pi} \int_{-\infty}^\infty e^{-i\phi k} F_{c_T} d\phi && (4.6) \\ &= e^{-\alpha k} \frac{1}{2\pi} \int_0^\infty e^{-i\phi k} F_{c_T} d\phi && (4.7) \end{aligned}$$

Substituting (6.1) and (6.2) into (6.3),

$$\begin{aligned} F_{c_T}(\phi) &= \int_{-\infty}^\infty e^{i\phi k} e^{\alpha k} e^{-rT} \int_k^\infty (e^{x_T} - e^k) f_T(x_T) dx_T dk && (4.8) \\ &= \int_{-\infty}^\infty e^{-rT} f_T(x_T) \int_{-\infty}^{x_T} (e^{x_T} + \alpha k) - e^{(\alpha+1)k)} e^{i\phi k} dk dx_T && (4.9) \\ &= \frac{e^{-rT} F_{C_T}(\phi - (\alpha+1)i)}{\alpha^2 + \alpha - \phi^2 + i(2\alpha+1)\phi} && (4.10) \end{aligned}$$

where $F_{C_T}(\phi)$ is the characteristic function of x_T, under Q:

$$
\begin{aligned}
F_{C_T}(\phi) &= e^{A(\phi)+B(\phi)+C(\phi)} \\
A(\phi) &= i\phi(x_0 + rT) \\
B(\phi) &= \frac{2\zeta(\phi(1-e^{-\psi(\phi)T}))V_0}{2\psi(\phi) - (\psi(\phi) - \gamma(\phi))(1 - e^{-\psi(\phi)T})} \\
C(\phi) &= -\frac{k\theta}{\sigma^2}\left[2log\left(\frac{2\psi(\phi) - (\psi(\phi) - \gamma(\phi))(1 - e^{-\psi(\phi)T})}{2\psi(\phi)}\right) + (\psi(\phi)) - \gamma(\phi))T\right] \\
\zeta(\phi) &= -\frac{1}{2}(\sigma^2 + i\phi) \\
\psi(\phi) &= \sqrt{\gamma(\phi)^2 - 2\sigma^2\zeta(\phi)} \\
\gamma(\phi) &= k - \rho\sigma\phi i
\end{aligned}
$$

4.2.2 Pricing with FFT

An discrete approximation of $C_T(k_u)$ is found as

$$
C_T(k_u) \approx \frac{e^{-\alpha k_u}}{\pi} \sum_{j=1} N e^{-i\frac{2\pi}{N}(j-1)(u-1)} e^{ibv_j} F_{c_T} \frac{\eta}{3}(3 + (-1)^j - \sigma_{j-1}) \tag{4.11}
$$

where,

$$
\begin{aligned}
v_j &= \eta(j-1) \\
\eta &= \frac{c}{N} \\
c &= 600 \\
N &= 4096 \\
b &= \frac{\pi}{\eta} \\
k_u &= -b + \frac{2b}{N}(u-1), u = 1, 2, ..., N+1
\end{aligned}
$$

And now we can apply FFT to the right part of (6.11).

4.3 Comparison: Accuracy and Efficiency

Now, to make a comparison of these three methods listed above, assume there is an call option with $\kappa = 3, \theta = 0.05, \sigma = 0.05, \rho = -0.9, r = 0.01, v_0 = 0.05, s_0 = 50, strike = 50, T = 0.5$.To make the time difference more stable and noticeable, we price the option by the three methods 100 times respectively, and this gives:

Method	Time(s)	Price($)
FFT	0.253321	3.1959
Riemann	0.575859	3.2662
Quadl	0.386235	3.2642

Compared with the other two numerical methods, FFT do tends to be faster. Actually here, we are not really utilizing the power of FFT yet. The real advantage of FFT is that it can be used to calculate the price of a set of options with different strike price. However, there is a considerable price discrepancy between the price given by the FFT method and the other two. FFT, in this application, suffers from one drawback, i.e., accuracy. The solutions produced are depend on the choice of α. So, for our calibration problem, we'll use quadl method as our pricing function to calculate the square difference of the market price and the model price.

Chapter 5

Calibration of Heston Model

So far, we've reach a point to calibrate the Heston Model with the market data. We'll that calibration is actually a least-square optimization problem. In this chapter, two approaches are discussed. Of course, it's all based on `quadl`, since we've found in previous chapter it give us a good balance between time efficiency and accuracy. A local optimization approach which turns out to be more practical since efficiency in terms of speed. Also, we included the global optimization approach here as an comparison. Throughout, we examine the merits and drawbacks of the routines, with reference to their accuracy and robustness, as well as to their complexity. In this chapter we will not consider the moneyness or time structure, since this chapter is mainly for a comparison about different optimization methods. Moneyness or time structure issue will be addressed in next chapter. For illustration purpose, we'll apply different optimization methods to the call options traded on day01 just in order to get a idea of the difference between these two methods.

5.1 A Least-Square Optimization Problem

A well documented and popular method of fitting models to observed data is to find a set of model parameters values that minimizes the square of the differences between the empirical values and the corresponding model values. For this project, this requires us to minimize the square difference of the option prices generated by each of our models and the option prices observed in the market.

Suppose a sample option data for N traded option prices. Let $\Psi_i, i = 1, ...N$ be the market price of the ith call option. Let model price of a call option be $\hat{\Psi}_i$.The cost function associated with optimization is the minimization of

$$\sum_{i=1}^{N}(\Psi_i - \hat{\Psi}_i)^2$$

subject to

$$2k\theta \geq \sigma^2$$

5.2 A Local Optimization Approach

Matlab has a built in function `lsqnonlin` for least-squares non-linear optimization.`lsqnonlin` requires the user-defined cost function to compute the vector-valued function. The output is then the set of parameters which minimizes the sum of squares. For our problem, the cost function would be:

```
function [cost]=cost1(x)
global strike; global T; global S; global r; global mrkprice;
for i=1:length(T)
cost(i)=mrkprice(i)-HestonCallQuad(x(2),x(3),x(4),x(5),x(1),r(i),
...,T(i),S(i),strike(i));
end
end
```

Importantly, this routine is a a local optimization scheme and is sensitive to the initial parameter estimates. A poorly chosen initial point might lead to wrong parameters.

We choose three sets of initial parameters and apply them to the call options traded on day01. This is only for illustration purpose to get a better understanding about the difference between these two methods and this give us:

CASE	v_0	κ	θ	σ	ρ	time(s)	square error
Initial 1	.05	1	.05	.05	$-.9$		
Result 1	.1000	14.6564	0.0666	0.3209	-0.9000	10.1905	37.4106
Initial 2	.15	5	.15	.10	$-.5$		
Result 2	0.1000	13.4476	0.0672	0.4253	-0.9000	10.7439	36.5121
Initial 3	.35	10	.25	.15	0		
Result 3	0.1000	19.4896	0.0674	0.3822	-0.9000	10.7912	30.2427

Surprisingly,our result shows that the estimation for parameters are actually quite stable regards to different initial point. This might change since we only got three different sets of initial point.

5.3 A global Optimization Approach: Genetic Algorithm

In the computer science field of artificial intelligence, genetic algorithm (GA) is a search heuristic that mimics the process of natural selection. This heuristic (also sometimes called a metaheuristic) is routinely used to generate useful solutions to optimization and search problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. We'll not give a detailed description of the algorithm here and we'll use the Matlab built in function `ga(fitnessfcn,nvars)` to find the local minimum of the objective function, `fitnessfcn`.

To utilize this function, we need construct a objective function first. Apparently, we can no longer use the cost function, however, we can change the cost function slightly and get the objective function

```
function cost=objct(x)
global mrkprice; global strike; global T; global S; global r;
for i=1:7
cost(i)=mrkprice(i)-HestonCallQuad(x(2),x(3),x(4),x(5),x(1),
...,r(i),T(i),S(i),strike(i));
end
cost=cost.*cost;
cost=sum(cost);
end
```

Since genetic algorithm is a global optimization approach there is need to feed it with initial value. Yet, we still do need the constraint. By applying the same constraint set, it give us:

CASE	v_0	κ	θ	σ	ρ	time(s)	square error
Initial 1	0.0643	1.1162	0.0811	0.2259	-0.3885	67.7254	7.7025

5.4 Comparison: Accuracy and Efficiency

Result from the previous section shows that, as a global optimization method, genetic algorithm is far more time consuming than local optimization method. However, it do give us a better set estimation of parameters. The square error from genetic algorithm is significantly smaller than the square error from local minimization method. In practice, one good way might be to use genetic algorithm to get a set of parameter value and then use it as initial value for local method in situation when speed is highly required.

Chapter 6

Calibration with Moneyness and Time Structure

In this chapter, Heston model would be calibrated by using the market data given. Since the data is tick data from the market and not formatted in a way convenient to carry out the calibration. Some preparation work need to be done before we can move to the calibration work. In calibration part, result is organize as two parts. One in term of Moneyness of option, the other in term of Time Structure. Start from the raw data to the organized result, every is done by Matlab, and to run the program on a laptop, it will take about 20 minutes.

6.1 Dealing with the data

Only data of call options are used to carry out the calibration. We have 14 sets of data from different trading days. For each option only the first trading price and corresponding data are used to carry out the calibration. Also, to perform the optimization, the data set needs to be larger than the number of parameters, so, at least 5 days are needed for each option. Here, we use data from all the 14 trading days, which means, for each option, there would be 14 item of data.

. **Filtering** This is for filtering the first trading record for each option. Meanwhile, at two records for an individual option in a single day should be satisfied, and this gives:

. **Matching** Once, we obtain the first trading record for each option on each day. We need to match option on difference trading days. Of course, this is done by matching Strike price and Expiration date for option record from different days, at last, 105 options match all the requirement and have record in all the 14 trading days. So, all calibration would be carried out on these 105 options.

6.2 Moneyness of an Option

Moneyness of an option indicates whether an option is worth exercising or not. Moneyness of an option at any given time depends on where the price of the underlying asset is at that point of time relative to the strike price. The following three terms are used to define the moneyness of an option:

An option is **ITM (in-the-money)** if on exercising the option, it would produce a cash inflow for the holder. Thus, call options are ITM at time t when the value of the price of the underlying exceeds the strike price $X_t > K$. On the other hand, put options are ITM when the price of the underlying is lower than the strike price, $X_t < K$.

An **OTM (out-of-the-money)** option is an opposite of an ITM option. A holder will not exercise the option when it is OTM. A call option is OTM when its strike price is greater than the price of the underlying and a put option is OTM when the price of the underlying is greater than the options strike price.

An **ATM (at-the-money)** is one in which the price of the underlying is equal to the strike price. It is at the stage where with any movement in the price of the underlying, the option will either become ITM or OTM.

The moneyness for call and put options is defined by:

$$M_t = \frac{K}{X_t}$$

where K is the strike price and X_t is the price of the underlying asset at time t.

Moneyness	Call	Put
< 0.94	Deep ITM	Deep OTM
$0.94 - 0.97$	ITM	OTM
$0.97 - 1.03$	ATM	ATM
$1.03 - 1.06$	OTM	ITM
> 1.06	Deep OTM	Deep ITM

6.3 Calibration in term of Moneyness

In this section, calibration result is organized in terms of Moneyness. And a brief comparison in terms of Moneyness:

Moneyness	v_0	κ	θ	σ	ρ	square error
DeepITM	0.0672 (0.0227)	6.4860 (8.2350)	0.1246 (0.0896)	0.4882 (0.1919)	-0.0453 (0.4069)	20.5670 (18.7765)
ITM	0.1212 (0.0227)	7.937 (8.2350)	0.0402 (0.0896)	0.428 (0.1919)	-0.6815 (0.4069)	25.7964 (18.7765)
ATM	0.0837 (0.0354)	13.3740 (10.0375)	0.0619 (0.0237)	0.3968 (0.2314)	-0.2941 (0.5044)	40.0862 (22.6076)
OTM	0.0825 (0.0243)	11.1411 (10.8849)	0.0603 (0.0460)	0.4730 (0.1548)	-0.0686 (0.4842)	27.4603 (11.2388)

6.4 Calibration in term of Time structure

In this section, calibration result is organized in two category. With a Days to expiration less than 60 days; with a Days to expiration larger than 60 days; then a brief comparison:

DayToEx(day)	v_0	κ	θ	σ	ρ	square error
< 60	0.0772 (0.0283)	8.3091 (9.7994)	0.1061 (0.0947)	0.5051 (0.2070)	-0.0653 (0.4434)	17.9827 (12.7396)
$60 - 180$	0.0641 (0.0224)	8.4637 (7.9012)	0.0983 (0.0410)	0.3870 (0.1520)	-0.1995 (0.4460)	40.8589 (24.0011)

Our calibration result shows that for options with short term the standard error for parameters are larger, but sum of square error is smaller.

Bibliography

[1] G. Bakshi, C. Cao, and Z. Chen, *Empirical performance of alternative option pricing models*, The Journal of Finance, 52 (1997), pp. 2003–2049.

[2] R. Bauer, *Fast calibration in the heston model*.

[3] T. Björk, *Arbitrage theory in continuous time*, Oxford university press, 2004.

[4] P. Carr and D. Madan, *Option valuation using the fast fourier transform*, Journal of computational finance, 2 (1999), pp. 61–73.

[5] E. K. Chong and S. H. Zak, *An introduction to optimization*, vol. 76, John Wiley & Sons, 2013.

[6] J. Gatheral, *The volatility surface: a practitioner's guide*, vol. 357, John Wiley & Sons, 2006.

[7] S. L. Heston, *A closed-form solution for options with stochastic volatility with applications to bond and currency options*, Review of financial studies, 6 (1993), pp. 327–343.

[8] S. Mikhailov and U. Nogel, *Hestons stochastic volatility model: Implementation, calibration and some extensions*, The Best of Wilmott, p. 401.

[9] N. Moodley, *The heston model: a practical approach with matlab code*, University of the Witwatersrand, (2005).

[10] M. Ncube, *Heston model calibration and simulation*, (2010).

[11] W. Poklewski-Koziell, *Stochastic volatility models: calibration, pricing and hedging*, PhD thesis, 2012.

[12] S. E. Shreve, *Stochastic calculus for finance II: Continuous-time models*, vol. 11, Springer, 2004.