Stable Local Volatility Function Model Calibration Using RBF Kernel and Regularization

by

Xichen Liu

A research paper
presented to the University of Waterloo
in partial fulfillment of the
requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Supervisor: Prof. Yuying Li

Waterloo, Ontario, Canada, 2013

Abstract

We investigate an optimization formulation using nonlinear least squares with regularization terms to ensure accuracy and stability in the local volatility model calibrating for option prices. By controlling regularization parameters in our objective function, we can balance calibrating errors and model complexity. In this investigation our local volatility function is represented by a radial basis function kernel. The radial basis function kernel is sufficiently complex to calibrate local volatility function correctly. However, its complexity may greatly reduce the stability of the model. Therefore, we wish the coefficient vector of the radial basis function kernel is sparse and the radial basis functions are simple. If so, the model can be greatly simplified, which can ensure the stability of the model. Therefore, we use regularization terms to reach this purpose. The model accuracy is controlled by the nonlinear least squares, while the model complexity is controlled by the regularization terms. We will evaluate the performance of our model based on S&P 500 market index option data. In detail, we illustrate the accuracy through the relative errors between the calibrated option prices and the given option prices, and the stability through calibrating the local volatility function by two sets of market option data on close dates. In addition, we demonstrate that the calibrated local volatility surface is similar to the observed implied volatility surface.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

For model calibration problems in derivatives markets, traders usually evaluate the model validation from two aspects: accuracy and stability of the model. According to the Ockham's razor, accuracy requires that the model calibration should be complex enough to match all the given market data, however, unfortunately stability requires the model as simple as possible. Therefore how to balance the accuracy and stability of the model is a significant and interesting topic in finance.

Among all financial models, local volatility function (LVF) model, without a doubt, plays an important role for option pricing. To describe the local volatility function model, we need to introduce the well-known Black-Scholes model (Black and Scholes, 1973). The Black-Scholes model can be described as

$$\frac{dS_t}{S_t} = (r - q)dt + \sigma dZ_t \tag{1.1}$$

where S_t is the stock price at time t, r is the risk free interest rate, q is a constant continuous dividend rate and σ is a constant volatility. Here Z_t is a standard Brownian motion.

We can easily observe that the Black-Scholes (B-S) model is simple, which indicates it should be very stable. However, in fact the volatility in real market is more complex. Therefore, traders often use implied volatility to represent the constant volatility in B-S model. Thus in order to evaluate the option, traders generally offer the implied volatility instead of the actual option price. From this, we can see the implied volatility is a very important standard.

The local volatility function model is just a simple extension to the B-S model. The general form of a local volatility model is described as follows:

$$\frac{dS_t}{S_t} = (r-q)dt + \sigma(S_t, t)dZ_t \tag{1.2}$$

where the local volatility function σ just depends on the underlying price S_t and time t. The model is attractive because there is only one random term in the model. Because of this, in a local volatility function model, option risk can be completely eliminated by setting some conditions including no arbitrage, continuous trading and no transaction cost. Therefore, we can get the famous Black-Scholes partial differential equation (PDE):

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2(S,t)S^2\frac{\partial^2 V}{\partial S^2} + (r-q)S\frac{\partial V}{\partial S} - rV = 0 \tag{1.3}$$

From the PDE, we can easily see that the option value V just depends on the stock price S and the time t. By solving the PDE, traders can price options based on the local volatility function model.

Due to advantages mentioned above, the local volatility function model has been extensively investigated. Some practical and interesting local volatility functions have been put forward. In [7], the local volatility function is represented by a cubic spline with a fixed number of spline knots and end conditions. But this model may lack stability and have unrealistic oscillations. In [8], the local volatility function is represented by a kernel spline. The objective function is a regularized optimization formulation, where the regularization term is the 1-norm of the coefficient vector for a kernel spline. Through minimizing the 1-norm of the coefficient vector, they can reach the purpose of minimizing the number of support vectors, which in general leads to good performance for the calibrating. Unfortunately this process is difficult to automate and the performance of the local volatility function are quite dependent on the training region. Because for the kernel spline, the training points have to be select manually and the calibration accuracy can be kept only for the region close to the training points.

In this study, we use a radial basis function kernel to represent the unknown local volatility function. The radial basis function kernel is complex enough to represent most functions correctly. Due to its complexity, the radial basis function kernel may lack stability for model calibrating. In the context of radial basis function calibrating, we require the coefficient vector of the radial basis function kernel to be positive. Moreover, the radial basis function kernel can be greatly simplified by minimizing the 1-norm of the coefficient vector and making the radial basis functions constant as much as possible. Therefore, the complexity of the model is controlled by two regularization parameters based on the above perspectives.

Chapter 2

Optimization Formulation

2.1 Local Volatility Function

For a fixed pair of strike K and maturity T, we can get the corresponding initial European option value through solving the PDE (1.3), where we assume the initial underlying price S_0 at time $t = 0$ is given. Let $V^0(K, T)$ denote the initial European option value, which only depends on the strike K and maturity T. It can be shown that the initial option price $V^0(K, T)$ satisfies the dual Black-Schloes equation:

$$\frac{\partial V^0}{\partial T} - \frac{1}{2}\sigma^2(K,T)K^2\frac{\partial^2 V^0}{\partial K^2} + (r-q)K\frac{\partial V^0}{\partial K} + qV^0 = 0 \tag{2.1}$$

where $\sigma(K, T)$ is the corresponding local volatility. See details in [1] and [10].

By changing the form of (2.1), we get the following form (2.2):

$$\sigma^2(K,T) = 2\frac{\frac{\partial V^0}{\partial T} + qV^0 + (r-q)K\frac{\partial V^0}{\partial K}}{K^2\frac{\partial^2 V^0}{\partial K^2}} \tag{2.2}$$

The local volatility can be uniquely determined by the initial option price function $V^0(K, T)$, when the right hand side of the formula is always nonnegative.

However, the market provides option prices only for a limited finite set of strikes and maturities. Assume that m initial market option price $\{\overline{V_j^0}\}_{j=1}^m$ corresponding to strike and maturity pairs $\{K_j, T_j\}, j = 1, ..., m.$ are provided. Thus the problem changes into calibrating a stable local volatility function $\sigma(S_t, t)$ such that the calibrated option prices match all the given market data.

In the context of support vector learning, radial basis function kernel is sufficiently complex that it can have a good performance to approximate any given function. Therefore, we use the RBF kernel in (2.3) as our local volatility function:

$$\sigma(K,T) = \sum_{j=1}^{m} a_j e^{-\frac{\|(K,T)-(K_j,T_j)\|_2^2}{b_j^2}} \tag{2.3}$$

We can rewrite the local volatility as a function of underlying price S an t, which is shown in (2.4)

$$\sigma(S,t) = \sum_{j=1}^{m} a_j e^{-\frac{\|(S,t)-(K_j,T_j)\|_2^2}{b_j^2}} \tag{2.4}$$

2.2 Finite Difference Method

There are several methods to price the initial option values, of which solving the B-S PDE using finite difference method is obviously an easy and efficient method. The key technique is using finite difference method to get an approximation for all differential terms such as $\frac{\partial V}{\partial t}$, $\frac{\partial^2 V}{\partial S^2}$ and $\frac{\partial V}{\partial S}$. To this end, we have to define a finite difference grid. For S, we define $S_i, i = 1, .., k$. which means different stock values and $\Delta S_i = S_{i+1} - S_i, i = 1, ..., k-1$.. For t, we define the time step $\Delta t = t^{n+1} - t^n$.

Therefore, we can get the following finite difference approximations to terms in the B-S PDE:

$$\begin{aligned} \left(\frac{\partial V}{\partial t}\right)_i^n &= \frac{V_i^{n+1} - V_i^n}{\Delta t} \\ \left(\frac{\partial^2 V}{\partial S^2}\right)_i^n &= \frac{\frac{V_{i+1}^n - V_i^n}{\Delta S_i} - \frac{V_i^n - V_{i-1}^n}{\Delta S_{i-1}}}{\frac{\Delta S_i + \Delta S_{i-1}}{2}} \end{aligned} \tag{2.5}$$

For $\frac{\partial V}{\partial S}$,

$$\begin{aligned} \left(\frac{\partial V}{\partial S}\right)_i^n &= \frac{V_{i+1}^n - V_{i-1}^n}{\Delta S_i + \Delta S_{i-1}} \\ \left(\frac{\partial V}{\partial S}\right)_i^n &= \frac{V_{i+1}^n - V_i^n}{\Delta S_i} \end{aligned} \tag{2.6}$$

where the first row is the central form and the second row is the forward form. Substituting all the discrete approximations into the B-S PDE, we get the following scheme after some algebra calculations:

$$V_i^{n+1} = V_i^n(1 - (a_i + b_i + r)\Delta t) + V_{i-1}^n \Delta t a_i + V_{i+1}^n \Delta t b_i \tag{2.7}$$

The values of a_i and b_i depend on the choice of central, forward.

For the central form:

$$\begin{aligned} a_i &= \frac{\sigma^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} - \frac{rS_i}{S_{i+1} - S_{i-1}} \\ b_i &= \frac{\sigma^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} - \frac{rS_i}{S_{i+1} - S_{i-1}} \end{aligned} \tag{2.8}$$

For the forward form:

$$\begin{aligned} a_i &= \frac{\sigma^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} \\ b_i &= \frac{\sigma^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} + \frac{rS_i}{S_{i+1} - S_i} \end{aligned} \tag{2.9}$$

For the central form, the convergence rate is $O(\Delta S^2)$, while for the forward form, the convergence rate is $O(\Delta S)$. In order to make the scheme accurate, we should use central form as much as possible. But sometimes when we use central form, it may lead to an unstable scheme. Having all coefficients positive is the precise mathematical requirement for a reasonable discretization for a hyperbolic problem. Luckily for the forward form, we can guarantee the positivity of the corresponding coefficients. Therefore, our technique implement is described as follows:

- For each time-step, we should calculate the coefficients of our scheme using central form.
- If all the coefficients are positive, we will choose the central form. If not, we will use the forward form.

Applying the scheme (2.7), we can get the initial option value. This method is called an explicit method. But the explicit method has a global truncated error $O(\Delta t)$. In order to increase the stability of our finite difference method, we will refer to some more complex schemes. Firstly, we will introduce the fully implicit method, whose scheme is shown as:

$$V_i^{n+1} = V_i^n + \Delta t[a_i(V_{i-1}^{n+1} - V_i^{n+1}) + b_i(V_{i+1}^{n+1} - V_i^{n+1}) - rV_i^{n+1}] \tag{2.10}$$

Unfortunately the fully implicit method has the same time truncated error as explicit method $O(\Delta t)$. However, by combining explicit method and fully implicit method, the Crank-Nicolson (C-N) method has a better time truncated error $O(\Delta t^2)$. The Crank-Nicolson uses the following scheme:

$$\begin{aligned} V_i^{n+1} =& V_i^n + \frac{\Delta t}{2}[a_i(V_{i-1}^{n+1} - V_i^{n+1}) + b_i(V_{i+1}^{n+1} - V_i^{n+1}) - rV_i^{n+1}] \\ &+ \frac{\Delta t}{2}[a_i(V_{i-1}^{n} - V_i^{n}) + b_i(V_{i+1}^{n} - V_i^{n}) - rV_i^{n}] \end{aligned} \tag{2.11}$$

The values of a_i and b_i are the same as the above definition.

However, in practice, we do not use C-N method directly, because unreasonable oscillations may occur in delta (V_S) and gamma (V_{SS}) . In order to keep delta and gamma smooth and 2nd order convergence, we will refer to the C-N Rannacher method, which requires taking two fully implicit timesteps and then using C-N timesteps after each rough initial state. See details in [11].

2.3 The Objective Function

In this section, we will propose a regularized optimization formulation for optimization calibration. We have defined the local volatility function (2.4). Through the finite difference method, we can get the initial option price $V_j(a, b)$ corresponding to the pair of the strike price K_j and maturity T_j based on the local volatility function (2.4), where a and b are vector forms of a_j and b_j. Suppose the initial market option price $\overline{V}_j$ corresponding to strike and maturity pair $\{K_j, T_j\}$ has been given by the market. Therefore, we can easily get an unconstrained nonlinear least square problem through our calibrated option prices and the given market prices:

$$\min \|V(a, b) - \overline{V}\|_2^2 \tag{2.12}$$

where $V(a, b)$ and $\overline{V}$ are the vector forms of $V_j(a, b)$ and $\overline{V}_j$. Since volatility should be nonnegative, we require the coefficient vector a of the radial basis function kernel to be nonnegative. Furthermore, in order to control the complexity of the model, we need to minimize the 1-norm of the coefficient vector a and make radial basis functions as simple as possible. Since b_j is the width of the corresponding radial basis function, we can get $b_j \geq 0$. Therefore, we can just treat it as nonnegative. We want all the b_j to be as large as possible, since as $b_j \longrightarrow +\infty$ the corresponding radial basis function is close to constant 1. Combining all the requirements mentioned above, we get the final objective function with regularization parameters:

$$\min_{a \geq 0, b \geq 0} \|V(a,b) - \overline{V}\|_2^2 - \lambda_b \sum_{j=1}^{m} log(b_j) + \lambda_a \sum_{j=1}^{m} a_j \tag{2.13}$$

where λ_a and λ_b are two regularization parameters.

Chapter 3

Computational Examples

3.1 Solving the regularized optimization problem

To solve the optimization problem (2.13), we use the Matlab built-in function fmincon. To apply fmincon, the gradient of the objective function must be given. In addition, the Hessian matrix of the objective function can greatly speed up the convergence of the optimization problem. But the new problem is that the cost of computing the Hessian matrix is too much. In order to solve this problem, we can use a good approximation to the Hessian and the cost of calculating the approximation can be accepted relatively.

For the regularization terms, we can easily calculate the corresponding gradient and Hessian matrix. So the key point is how to calculate the gradient and approximate the Hessian matrix of $\|V(a,b)-\overline{V}\|_2^2$.

Firstly, we need to define some formulas to simplify our mathematical expression.

$$\begin{aligned} F(x) &= V(a,b)-\overline{V} \\ f(x) &= \|V(a,b)-\overline{V}\|_2^2 \end{aligned} \tag{3.1}$$

where $x=[a,b]$ (x is the vector form of a and b). Through some mathematical manipulation, we can get the gradient of $f(x)$ is

$$\nabla f(x) = 2J(x)^T F(x) \tag{3.2}$$

where $J(x)$ is the Jacobian matrix of $F(x)$. The Hessian matrix of $f(x)$ is

$$2J(x)^T J(x) + 2\sum_{i=1}^{m} \nabla^2 F_i(x) F_i(x) \tag{3.3}$$

where $F_i(x)$ is the ith entry of $F(x)$, and $\nabla^2 F_i(x)$ is the Hessian matrix of $F_i(x)$. Since all the terms $F_i(x)$ are likely small in magnitude when x is close to s minimizer, we can just use the following simple form to approximate the Hessian matrix of $f(x)$:

$$\nabla^2 f(x) \approx 2J(x)^T J(x) \tag{3.4}$$

The default algorithm for fmincon function is trust-region-reflective, which is an interior point trust region method for bound constrained problems. See details in [5] and [6].

3.2 Computing a good starting point

The optimization problem (2.13) is nonconvex. For any given starting point, we can find a local minimizer through optimization process, but it is hard to determine whether it is a global minimizer or it is sufficiently close to the global minimizer. Therefore, a good choice of the starting point a_0 and b_0 can increase the probability of finding the global minimizer and speed up the convergence rate of the method. In addition, the computational cost of the optimization algorithm can be greatly reduced with a good starting point. In local volatility calibration, it is reasonable to determine the value of a and b such that the corresponding volatility surface given by a and b is close to the observed implied volatility surface as much as possible.

We assume that m implied volatilities $\{\overline{\sigma}_i\}_{i=1}^m$ corresponding to the pairs of strike and maturity $\{(K_i, T_i)\}$ are given by the market data. Combing our local volatility function (2.3), it is reasonable to determine an initial point by solving the following nonlinear least squares problem (3.5):

$$\min_{a\geq 0, b\geq 0} \sum_{i=1}^{m} \left((\sum_{j=1}^{m} a_j e^{-\frac{\|(K_i,T_i)-(K_j,T_j)\|_2^2}{b_j^2}}) - \overline{\sigma}_i \right)^2 \tag{3.5}$$

However, this problem is almost as complex as our proposed optimization problem. Here I give two simple ways to get a good approximation for the starting point.

The first way: in order to make our model stable, the value of b_j should be as large as possible. So if we substitute some large values for b_j, our optimization problem can be simplified into the following bounded least square problem:

$$\min_{a\geq 0} \sum_{i=1}^{m} \left((\sum_{j=1}^{m} a_j e^{-\frac{\|(K_i,T_i)-(K_j,T_j)\|_2^2}{b_j^2}}) - \overline{\sigma}_i \right)^2 \tag{3.6}$$

where b is a constant vector with sufficient large positive numbers,e.g. $b_j = 100$.

The second way is similar to the first in spirit. If b is a sufficient large positive vector, the coefficient matrix for the above optimization problem is close to the matrix whose entries are all 1. The solution for this problem is just that the sum of a_j equals to the average of all $\overline{\sigma}_i$. If we require all a_j are equivalent, we can easily get all the values of a_j.

3.3 Calibration from S&P 500 index option data

In order to check how our calibration approach performs with the real market data, we implement the local volatility model using the S&P 500 market option data. I use the same data sets in [1] [7] and [8]. Table 3.1 represents implied volatilities one day in Oct 1995. On that day, the corresponding S&P 500 index value S_0=\$590, interest rate r=6%, and dividend rate q=2.62%. The corresponding European call option prices are listed in Table 3.2.

Maturity\Strike	85%	90%	95%	100%	105%	110%	115%	120%
0.695	0.172	0.157	0.144	0.133	0.118	0.104	0.100	0.101
1	0.171	0.159	0.150	0.138	0.128	0.115	0.107	0.103
1.5	0.169	0.160	0.151	0.142	0.133	0.124	0.119	0.113

Table 3.1: Implied volatility for Oct 95 S&P 500 index options with strikes in % of the spot price.

Maturity\Strike	85%	90%	95%	100%	105%	110%	115%	120%
0.695	101.9	76.26	52.76	32.75	16.47	6.02	1.93	0.62
1	108	83.6	61.55	41.57	25.41	12.75	5.5	2.13
1.5	117.2	94.37	73.14	53.97	37.33	23.68	14.3	7.65

Table 3.2: European call prices on Oct 95 S&P 500 index options with strikes in % of the spot price.

Figure 3.1 shows the observed implied volatility surface given by Table 3.1 :

Firstly, we should calculate the starting point to analyze our problem: we choose 100 as the values of all b_j. By those two ways introduced in Section 3.2, we can calculate the corresponding a_j for each way based on Table 3.1. In order to determine the better starting point, we test the performance of these two starting points based on Case 1.

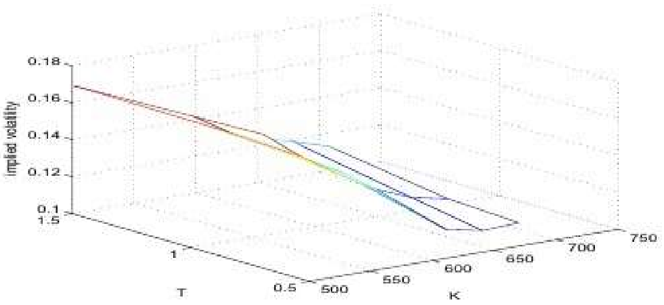

Figure 3.1: Implied volatility for S&P 500 index option market data on Oct 1995.

Using fmincon function, after each iteration, the change of the objective function value is decreasing. Therefore, through balancing the optimization cost and the accuracy of the optimizer, we require that it just takes 30 iterations to get the solution. For the first starting point, the initial objective function value is 59.7967, but after 30 iterations, the objective function value decreases to 4.7706. While for the second starting point, the initial objective function value is 1444.68, but after 30 iterations, the objective function value decreases to 4.34125. Moreover, the second way is easier. Both of these suggest that the second starting point is a better choice for this case.

Through extensive testing, the starting points given by these two ways perform similarly. Due to the simplicity of the second way, we calculate our starting point by that way.

Next, we investigate the influence of λ_a and λ_b on calibrating. In order to check how the calibrated local volatility surface performs with different λ_a and λ_b , we choose the value of λ_a and λ_b in the $4 * 4$ grids, described in Table 3.3.

Value of λ_a and λ_b	$\lambda_a = 0.1$	$\lambda_a = 1$	$\lambda_a = 10$	$\lambda_a = 100$
$\lambda_b = 0.1$	Case 1	Case 2	Case 3	Case 4
$\lambda_b = 1$	Case 5	Case 6	Case 7	Case 8
$\lambda_b = 10$	Case 9	Case 10	Case 11	Case 12
$\lambda_b = 100$	Case 13	Case 14	Case 15	Case 16

Table 3.3: Cases for different choices of λ_a and λ_b

Through our model and optimization process, we get the corresponding calibrated local volatility surfaces. We graph them in Figure 3.2 and Figure 3.3.

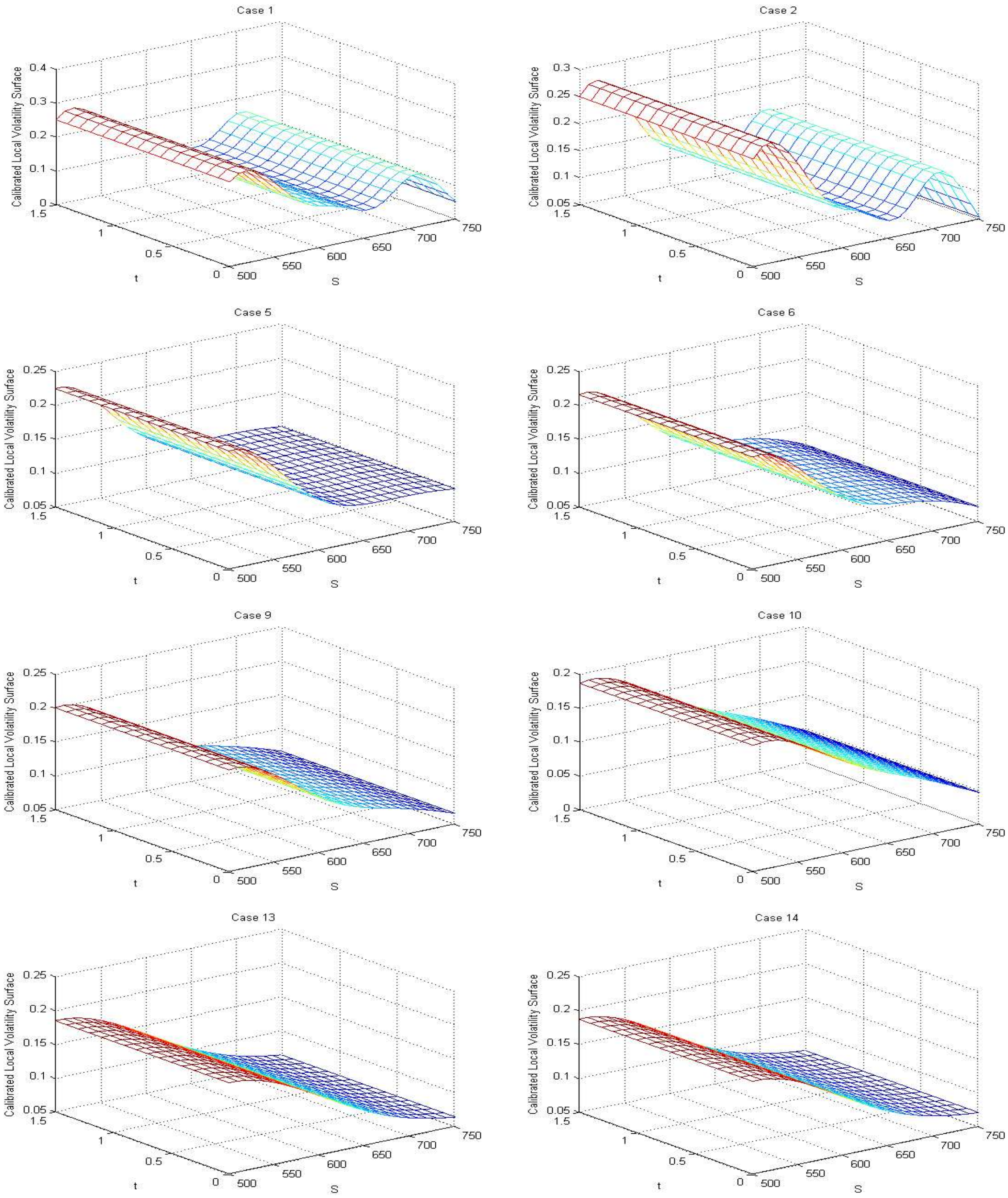

Figure 3.2: Calibrated local volatility surfaces for all the cases where $\lambda_a = 0.1$ and $\lambda_a = 1$

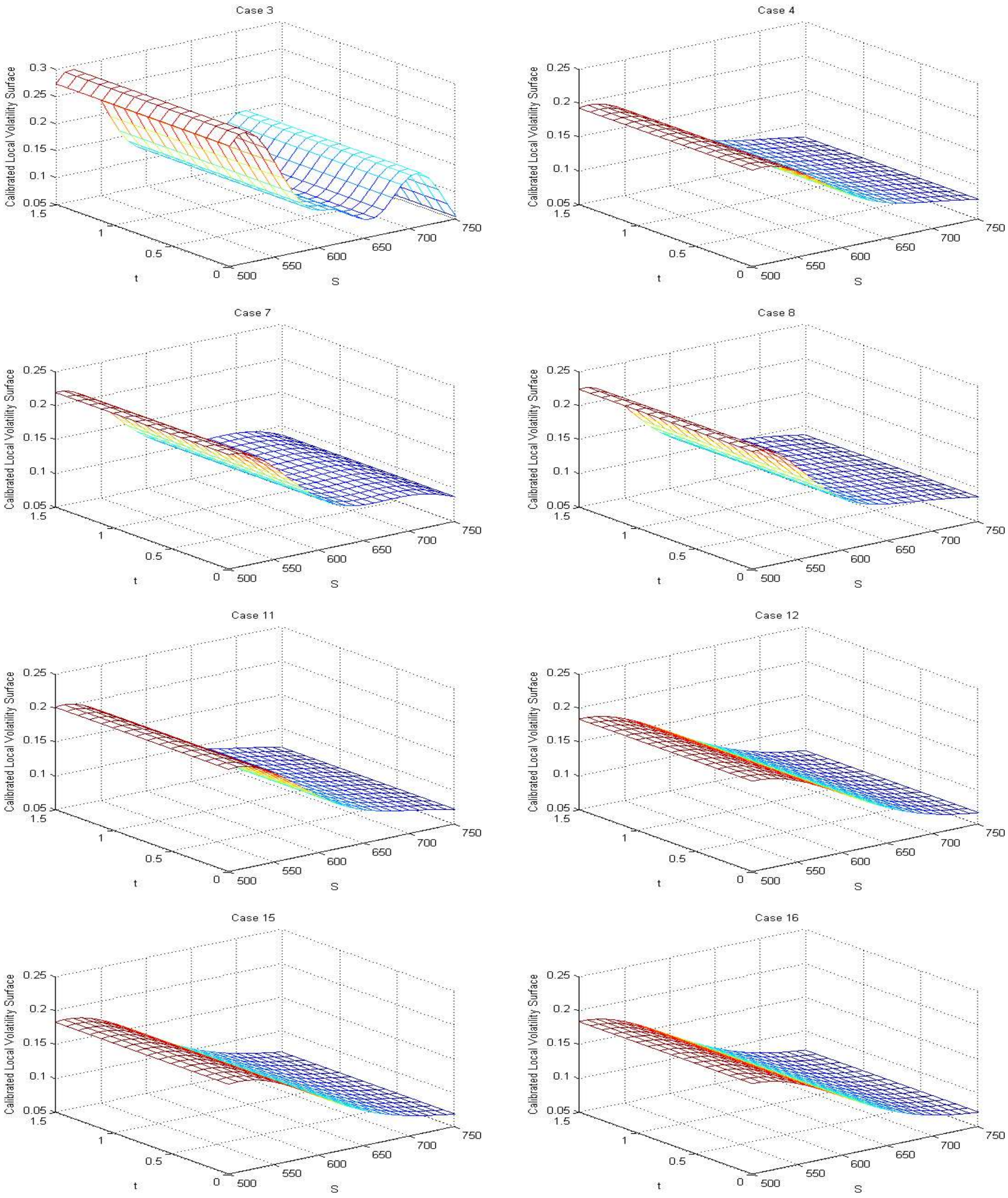

Figure 3.3: Calibrated local volatility surfaces for all the cases where $\lambda_a = 10$ and $\lambda_a = 100$

In spite of its simplicity and attractive properties, the local volatility function model often meets some criticisms in the real derivatives markets. There are two main criticisms: the first is the calibrated local volatility surface often have unreasonable oscillations, which you can refer to [7], and the other one is the calibrated local volatility surface usually has large changes for the options with a small maturity. The second phenomenon is reasonable. For the options with a small maturity, its option value is nearly the payoff of the option, so in order to change a small value for the option price, there must be a large change for the implied volatility.

From Figure 3.2 and Figure 3.3, we observe that the local volatility surface is very smooth for all the cases. With the value of λ_a and λ_b increasing, we can find the local volatility surface is more flat and similar to the implied volatility surface in shape. However, if we choose larger values for λ_a and λ_b , it will influence our model accuracy greatly. It seems that the pair of $\lambda_a = 100$ and $\lambda_b = 100$ is an appropriate choice here, and we will use these two values in the following calibrations.

In order to check the accuracy of our model, we can use the relative price errors, which is defined as $\frac{V(a,b)-\overline{V}}{\overline{V}}$ (in %). The relative price errors are shown in Table 3.4

Maturity\Strike	85%	90%	95%	100%	105%	110%	115%	120%
0.695	-0.71	-0.09	1.47	4.04	14.20	40.83	54.59	10.59
1	-0.91	-0.51	-0.37	1.10	2.38	9.77	15.11	2.90
1.5	-1.21	-1.22	-1.23	-1.49	-2.21	-3.48	-10.62	-20.07

Table 3.4: Relative calibration error in % for S&P 500 index options on Oct 1995 for case 16

The calibration error for most option are within ±5%. However, for some specific options which are out-of-the money options, the errors are quite large, because these prices are relatively smaller. Thus the contribution of the corresponding squared errors to the objective function becomes relatively small. Thus if we choose larger weights on these terms, the calibration errors for out-of-the money call options can be decreased. Basically jump model has been suggested in finance to explain the error for out-of-the-money option.

Therefore we will investigate the calibration with individual weights. Seeing from Table 3.4, we observe that the range of error is from 0 to 60%. Therefore, we can change the weights by setting the individual weights as (3.7)

$$w_j = \begin{cases} 40 & error > 40\% \\ 20 & 20\% < error < 40\% \\ 5 & 5\% < error < 20\% \\ 1 & otherwise \end{cases} \tag{3.7}$$

Therefore, we get the corresponding local volatility surface shown in Figure 3.4.

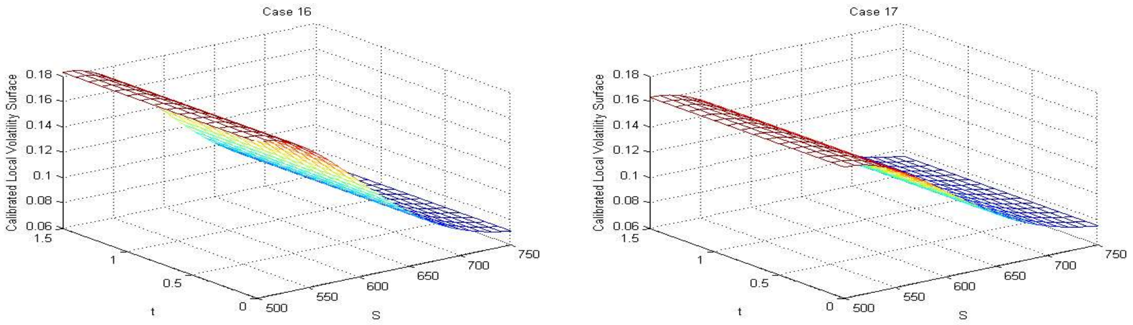

Figure 3.4: Calibrated local volatility for S&P 500 index option market data on Oct 1995 for case 16. Left plot is for uniform weights. Right plot is using individual weights

The corresponding relative price errors are shown in Table 3.5:

Maturity\Strike	85%	90%	95%	100%	105%	110%	115%	120%
0.695	-1.59	-1.91	-1.80	-1.28	5.87	27.90	39.82	2.85
1	-2.08	-2.57	-3.62	-3.67	-4.02	1.58	6.40	-2.92
1.5	-2.62	-3.40	-4.34	-5.62	-7.30	-9.19	-15.85	-23.68

Table 3.5: Relative calibration error in % for S&P 500 index options on Oct 1995 with individual weights

The calibrated local volatility for specific maturities are shown in Figure 3.9 .

Comparing Figure 3.4 with Figure 3.1, we can find the calibrated local volatility surface using individual weights is higher for the out-of-the-money option than that using uniform weights.

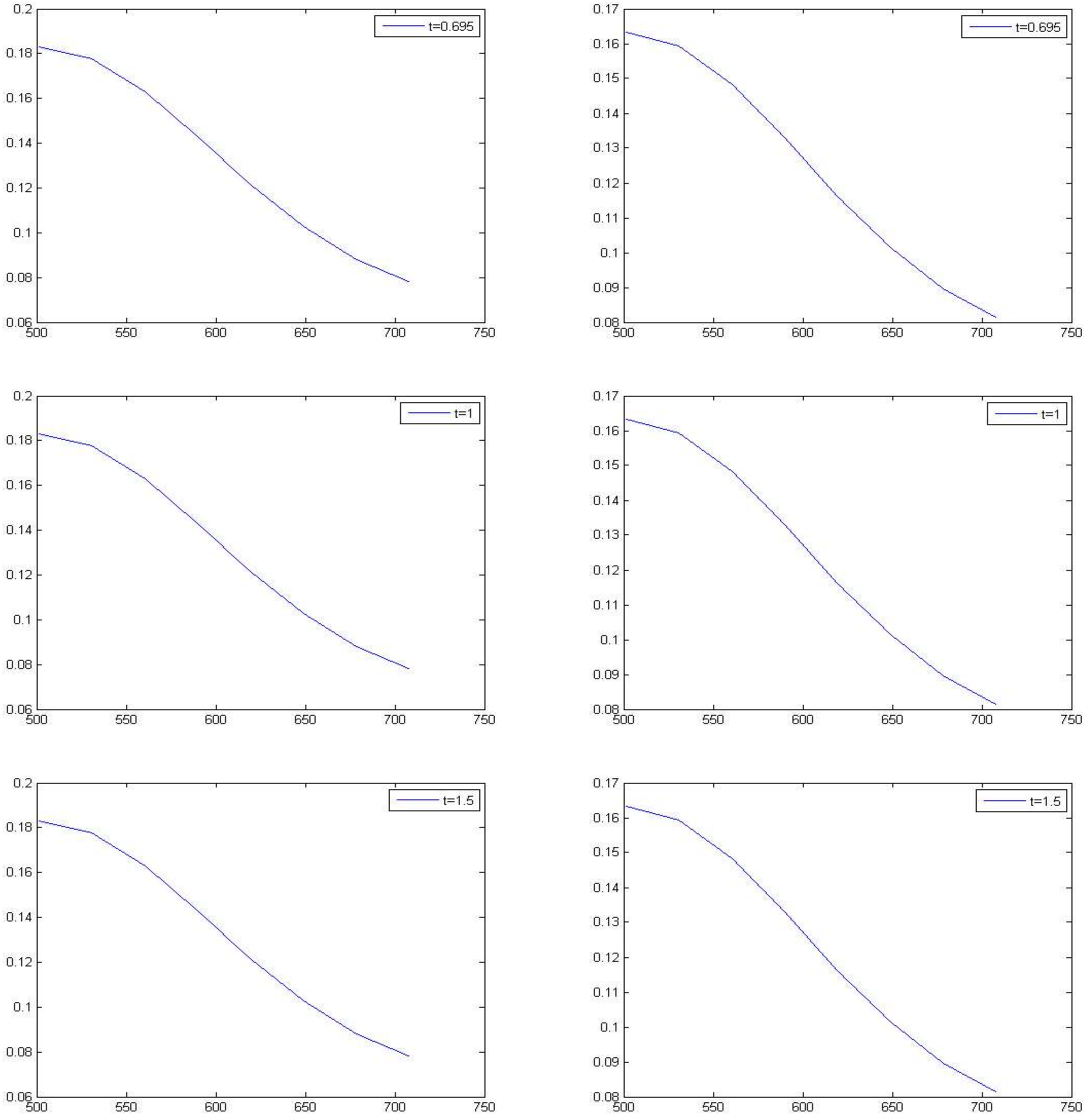

Figure 3.5: Calibrated local volatility for S&P 500 index options on Oct 1995. The left plot is using uniform weights. The right plot is using individual weights

3.4 Stability of Proposed LVF Model

We investigate the stability of the local volatility function model from two perspectives:

- A stable calibrated LVF model should return model prices similar to market data on nearby dates. Specifically, we calibrate a local volatility model on one particular day and compare the calibrated implied volatility surface to observed market implied volatility on nearby dates.

- A stable LVF model should yield similar local volatility surfaces for option data on nearby dates.

Therefore, we consider two sets of S&P 500 options data from two close dates, March 2, 2004 and April 5, 2004 respectively. On March 2, 2004, the index value $S_0 = \$1149.1$, while on April 5, 2004, the index value $S_0 = \$1150.57$. The other parameters for the two options are the same: interest rate $r = 1\%$ and dividend yield $q = 1.6\%$. Table 5 and 6 present the implied volatilities and the corresponding European call prices on March 2, 2004. Table 7 and 8 present the implied volatilities and the corresponding European call prices on April 5, 2004. (Using the same data sets in [1] [7] and [8])

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.58	0.197	0.1872	0.1645	0.1588	0.1538	0.1398	0.1323	0.1257
0.84	0.194	0.1801	0.1709	0.1595	0.1576	0.1448	0.1344	0.1324
1.34	0.1976	0.1908	0.1782	0.1725	0.1649	0.1577	0.1503	0.1402

Table 3.6: Implied volatility for S&P 500 index options on March 2, 2004

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.58	141	120.4	81	65	50.9	26.9	12.7	5
0.84	148.7	127.1	93	75	62.2	37.4	20	10.9
1.34	164.2	145.8	111.8	96.4	81	57.6	38.6	23

Table 3.7: European call prices for S&P 500 index options on March 2, 2004

We use the data given by the Table 3.8 and Table 3.6 to get the corresponding observed implied volatility surfaces in Figure 3.6.

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.5	0.1952	0.1852	0.1597	0.1582	0.1471	0.1305	0.1228	0.1164
1	0.201	0.1936	0.1797	0.1689	0.1628	0.152	0.1473	0.1394
1.25	0.2097	0.196	0.1894	0.1785	0.1776	0.1673	0.1584	0.1511

Table 3.8: Implied volatility for S&P 500 index options on April 5, 2004

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.5	138.9	117.9	77.2	62.2	46	21.7	8.8	2.8
1	157.1	138.1	103	85.2	70.6	45.9	29.2	16.3
1.25	167.7	146.4	115.2	97.3	85.3	60.2	40.3	25.6

Table 3.9: European call prices for S&P 500 index options on April 5, 2004

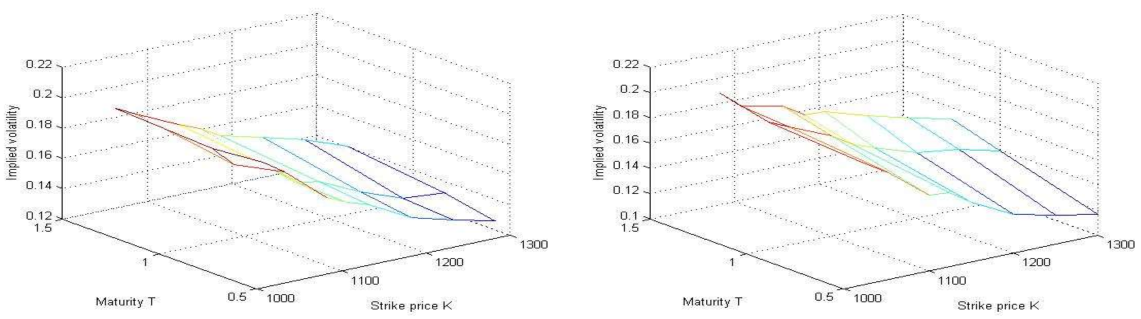

Figure 3.6: Implied volatility for S&P 500 index option market data. Left plot is for market option data on March 2, 2004. Right plot is for market option data on April 5, 2004.

3.4.1 The first perspective of stability

In order to check the first perspective, we calibrate our local volatility function model based on the market data on March 2, 2004 as the process described in section 3.3. Then we apply the calibrated local volatility function model directly to the market data on April 5, 2004 to evaluate the performance of our local volatility function model. In the process of applying the calibrated LVF model to the market data on April 5, we need to care that the variable t is the exact time, not the relative time.

For the choices of λ_a and λ_b, we can refer to the conclusion shown in section 3.3: $\lambda_a = 100$ and $\lambda_b = 100$.

Therefore we get the corresponding calibrated implied volatility surfaces shown in Figure 3.7.

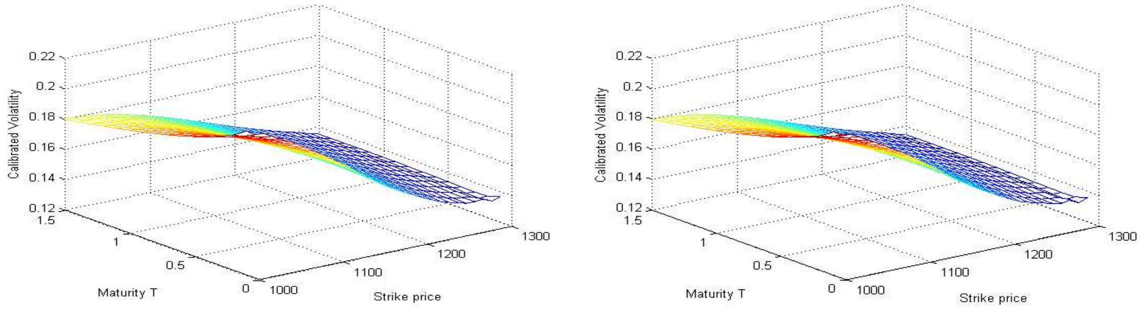

Figure 3.7: Calibrated implied volatility surface for S&P 500 index option market data. Left plot is for market option data on March 2, 2004. Right plot is for market option data on April 5, 2004.

The relative price errors for these two options are shown in Table 3.10 and 3.11.

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.58	-0.47	0.73	5.88	7.03	7.65	14.28	17.51	26.49
0.84	-1.09	1.54	2.32	5.96	4.38	7.69	12.78	4.69
1.34	-4.46	-3.85	-3.04	-3.06	-2.06	-5.50	-9.63	-9.57

Table 3.10: Relative price errors in % for S&P 500 index options on March 2, 2004.

From Figure 3.7 Table 3.10 and Table 17, we can observe that these two calibrated implied volatility surface are quite similar and the relative price errors for the option data

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.5	0.11	1.34	7.69	7.17	12.38	27.89	43.61	75.94
1	-3.51	-2.98	-2.23	0.00	0.14	-0.36	-7.10	-9.77
1.25	-6.79	-4.82	-7.03	-5.49	-8.93	-12.56	-17.62	-24.21

Table 3.11: Relative price errors in % for S&P 500 index options on April 5, 2004.

on April 5, 2004 are almost small. Therefore, we can conclude that our calibrated local volatility function model will yield similar implied volatility surface and relatively small relative price errors when it is applied on the market data on nearby date, which perfectly proves the first perspective of stability.

3.4.2 The second perspective of stability

Now we consider the second perspective of stability. We separately calibrate two local volatility function models based on the two sets of market data on nearby dates (March 2, 2004 and April 5, 2004).

The relative price errors for these two options are shown in Table 3.12 and 3.13.

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.58	-0.47	0.73	5.88	7.03	7.65	14.28	17.51	26.49
0.84	-1.09	1.54	2.32	5.96	4.38	7.69	12.78	4.69
1.34	-4.46	-3.85	-3.04	-3.06	-2.06	-5.50	-9.63	-9.57

Table 3.12: Relative price errors in % for S&P 500 index options on March 2, 2004.

Maturity\Strike	1025	1050	1100	1125	1150	1200	1250	1300
0.5	1.94	3.53	10.13	9.22	13.88	30.11	58.58	129.74
1	-1.74	-0.86	0.40	2.84	3.17	4.58	3.81	11.57
1.25	-5.19	-2.80	-4.45	-2.58	-5.78	-7.63	-7.94	-7.84

Table 3.13: Relative price errors in % for S&P 500 index options on April 5, 2004.

Figure 3.8 presents the corresponding calibrated local volatility surfaces.

Figure 3.9 graphs the corresponding calibrated local volatility for specific time.

From Table 3.12 and Table 3.13, we can conclude both of the calibrated local volatility function models are stable. From Figure 3.8 and Figure 3.9, we can observe that these two

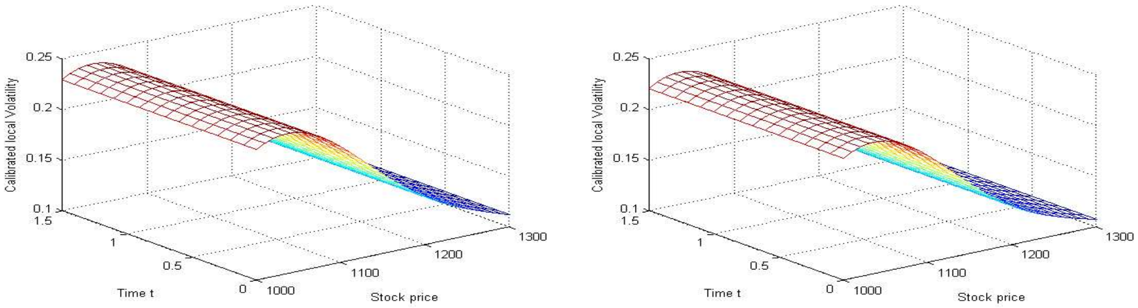

Figure 3.8: Calibrated local volatility surface for S&P 500 index option market data. Left plot is based on market option data on March 2, 2004. Right plot is based on market option data on April 5, 2004.

calibrated local volatility function models yield similar local volatility surfaces. Therefore, we have illustrated the stability of our model from the second perspective. In addition, comparing Figure 3.8 with Figure 3.6, we can see that these two calibrated volatility surfaces are similar to the observed implied volatility surface.

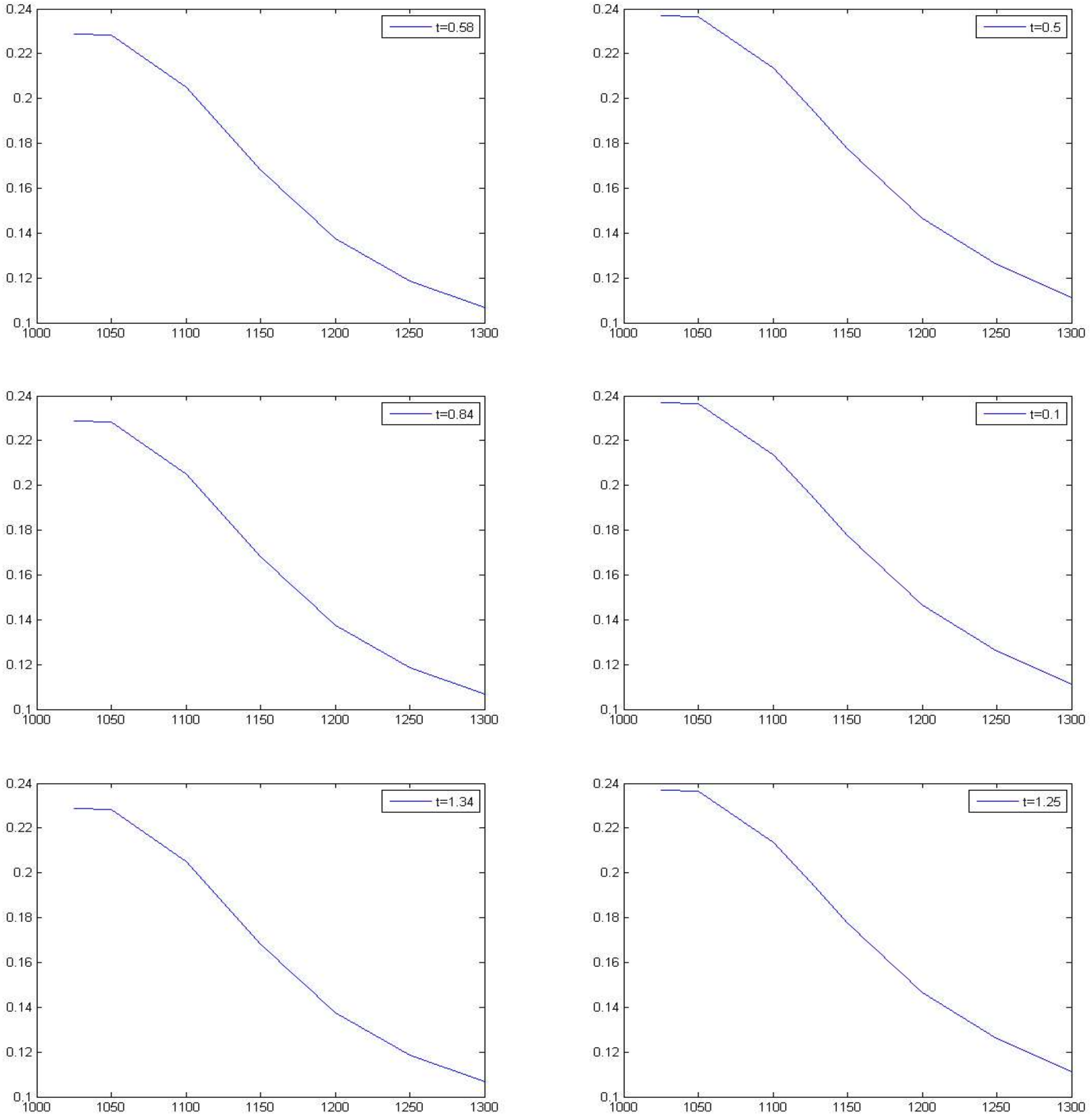

Figure 3.9: Calibrated local volatility for S&P 500 index options. The left plot is for the option market data on March 2,2004. The right plot is for the option market data on April 5, 2004. Uniform weights are used.

Chapter 4

Concluding Remarks

In this research, we investigate an optimization formulation using nonlinear least square with regularization terms to ensure the accuracy and stability of the local volatility function model calibrating for option prices. Our local volatility function is represented by the radial basis function kernel. The complexity of our model is controlled by the minimizing the 1-norm of the coefficient vector of the radial basis function kernel and choosing sufficient large kernel width parameters to make our radial basis function close to 1.

Based on preliminary computational results, we can make the following observations:

- The effect of regularization parameters λ_a and λ_b on the calibrated local volatility function model. From Figure 3.2 and 3.3, we find the local volatility surface is most flat for the case $\lambda_a = 100$ and $\lambda_b = 100$. However, if we choose larger values for λ_a and λ_b , it will greatly influence our model calibrating accuracy. Therefore, we should choose proper large values for λ_a and λ_b in order to ensure the stability of our model.

- Choosing the starting point. Because our model is a nonconvex function, the ideal starting point solved by the optimization problem (3.5) is as complex as our proposed regularization optimization formulation. Therefore, we have put forward two easy ways to get an approximation of the ideal starting point. Through testing, we find that both of these two ways return good solutions. In order to save the cost, we will choose the easier way to get the starting point.

- Calibration errors. We observe that the calibration relative errors are mostly within 5%. For some out-of-the-money options, the errors are large; because these prices are relatively smaller which leads to the small contribution of the corresponding squared errors to the objective function. If we choose larger weights on these options, the calibration errors for out-of-the-money call option can be decreased. We can observe this phenomenon in Table 3.5. And it can also be proved by Figure 3.4 that the calibrated local volatility surface using individual weights is higher for the out-of-the-money option than that using uniform weights..

Overall, we can conclude our calibrated local volatility model performs reasonably well. The accuracy of our model calibration is demonstrated by the small relative errors between the calibrated option price and the given option price. The stability of our model calibration is demonstrated by the facts that the calibrated local volatility functions from option data on close dates are similar and the calibrated local volatility function model gives back quite similar calibrated local volatility surfaces on nearby date. In addition, we can observe that the calibrated local volatility surface is similar to the observed implied volatility surface.

In this paper, we do not choose any training point for our local volatility model. Therefore, our calibrated LVF model seems to be accurate and stable only near the given pairs of strike price and maturity.

References

[1] L. ANDERSEN and R. BROTHERTON-RATCLIFFE. The equity option volatility smile: An implicit finite-difference approach. *Journal of Computational Finance*, 2:5–37, 1997.

[2] Leif Andersen and Jesper Andreasen. Jump-diffusion processes: Volatility smile fitting and numerical methods for option pricing. *Review of Derivatives Research*, 4(3):231–262, 2000.

[3] Guy Barles, Ch Daher, and Marc Romano. Convergence of numerical schemes for parabolic equations arising in finance theory. *Mathematical models and methods in applied Sciences*, 5(01):125–143, 1995.

[4] Thomas F Coleman, Changhong He, and Yuying Li. Calibrating volatility function bounds for an uncertain volatility model. *Journal of Computational Finance*, 13(4):63, 2010.

[5] Thomas F Coleman and Yuying Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on optimization*, 6(2):418–445, 1996.

[6] Thomas F Coleman and Yuying Li. A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.

[7] Thomas F Coleman, Yuying Li, and Arun Verma. Reconstructing the unknown local volatility function. In *Quantitative analysis in financial markets: collected papers of the New York University Mathematical Finance Seminar*, volume 2, page 192, 2001.

[8] Thomas F Coleman, Yuying Li, and Cheng Wang. Stable local volatility function calibration using spline kernel. *Computational Optimization and Applications*, pages 1–28, 2013.

[9] Thomas F Coleman and Arun Verma. Structure and efficient jacobian calculation. Technical report, Cornell University, 1996.

[10] B. DUPIRE. Pricing with a smile. *Risk*, 1:18–20, 1994.

[11] Peter Forsyth and Ken Verzal. *Numeric Computation for Financial Modelling*. 2011.

[12] Peter A Forsyth and Kenneth R Vetzal. Numerical methods for nonlinear pdes in finance. In *Handbook of Computational Finance*, pages 503–528. Springer, 2012.

[13] J Glover and MM Ali. Using radial basis functions to construct local volatility surfaces. *Applied Mathematics and Computation*, 217(9):4834–4839, 2011.

[14] Raul Kangro and Roy Nicolaides. Far field boundary conditions for black–scholes equations. *SIAM Journal on Numerical Analysis*, 38(4):1357–1368, 2000.

[15] Steven G Kou. A jump-diffusion model for option pricing. *Management science*, 48(8):1086–1101, 2002.

[16] Greg Orosi. Improved implementation of local volatility and its application to s&p 500 index options. *The Journal of Derivatives*, 17(3):53–64, 2010.

[17] David M Pooley, Peter A Forsyth, and Ken R Vetzal. Numerical convergence properties of option pricing pdes with uncertain volatility. *IMA Journal of Numerical Analysis*, 23(2):241–267, 2003.

[18] R Zvan, KR Vetzal, and PA Forsyth. Pde methods for pricing barrier options. *Journal of Economic Dynamics and Control*, 24(11):1563–1590, 2000.