Application of Smoothing Techniques to Implied Volatility

A Master Thesis Presented

by

Zeng, Jin

(184784)

to

Prof. Dr. Wolfgang Härdle

CASE - Center of Applied Statistics and Economics

Humboldt University, Berlin

in partial fulfillment of the requirements

for the degree of

Master of Arts

Berlin, September 14, 2005

Declaration of Authorship

I hereby confirm that I have authored this master thesis independently and without use of others than the indicated resources. All passages, which are literally or in general matter taken out of publications or other resources, are marked as such.

Zeng, Jin

Berlin, September 14, 2005

Acknowledgment

I would like to thank Professor Dr. Wolfgang Härdle for giving me the opportunity and motivation to write this thesis.

I'm especially indebted to Michal Benko for his excellent guidance all the time.

Furthermore I'm also grateful to my parents and my husband, without their support it would be impossible to finish this work.

Abstract

Implied volatility is an important element in risk management and option pricing. Black-Scholes model assumes a constant volatility, however, the evidence from financial market shows that the volatility is not constant but change with strike and time to maturity. In this paper, the time to maturity is fixed and we will construct the implied volatility function of strike or moneyness. We can use regression method for estimation, but the data from financial market contains some noise and we need to apply smoothing techniques to estimate this implied volatility function. The standard non- and semi-parametric regression methods don't guarantee the resulting IV functions are arbitrage free, so we will insert our estimation result to Black and Scholes model and calculate the state price density (SPD). In a Black-Scholes model it is lognormal distribution with constant volatility parameter. In practice as volatility changes the distribution deviates from log-normality. We estimate volatilities and SPDs using EUREX option data on the DAX index by using different smoothing techniques. Our estimation will be carried out through the strike direction and moneyness direction. We will briefly introduce Local polynomials as one method. The most important smoothing techniques we will applied in this paper is B-splines, with the usage of roughness penalty, which allows a flexible choice on the degree of smoothness, and is promising for future research work on the arbitrage free constraint of implied volatility.

Keywords: implied volatility, state price density, B-splines, roughness penalty, linear differential operator

Contents

1 Introduction 11

2 The Implied Volatility and SPD 15

2.1 Statistical Properties of Implied Volatility 15

2.2 Calculation the Implied Volatility 16

2.3 State Price Density 17

2.4 SPD estimation based on IV smile 20

2.5 The Quantlet spdcal and spdcalm 23

2.6 Arbitrage Free Constraint 24

3 Nadaraya-Watson and Local Polynomial Estimator 26

4 Smoothing IV With Functional Data Approach 33

4.1 Basic Setup 33

4.2 The Basis Function Approach 34

4.3 B-spline Bases 35

4.4 Estimation of Coefficients 36

4.5 Choice of the Smoothing Parameter 37

4.6 The Quantlet Bfacv and Bfagcv 39

5 The Implemention of Penalized B-splines on Financial Market Data 42

5.1 Construction of the Model 42

5.2 Volatility Smile as Function of Strike 43
5.3 Multiple Observations 55
5.4 Volatility Smile as Function of Moneyness I 58
5.5 Volatility Smile as Function of Moneyness II 61
5.6 Conclusions 64

6 Summary and Outlooks 65

7 Appendix 66

7.1 XploRe quantlet lists 66
7.1.1 Statistical Tools 66
7.1.2 Implemention Examples 71
7.2 Proofs 76
7.2.1 Convexity of Call Price Function 76
7.2.2 No Arbitrage Bounds of IV Smile 77

List of Figures

1.1 DAX option on 2003-02-25, Implied volatility with respect to strike and maturity . . . 13

2.1 DAX option on 25th Feb, 2003, $\tau = 0.14167$. Top panel: IV VS strike; bottom: IV vs moneyness . . . 18

3.1 IV smile with strike, DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: Bandwidth=150, lower panel: Bandwidth=250. Nadaraya-Watson estimator is used. . . . 29

3.2 IV smile with moneyness, DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: Bandwidth=0.05, lower panel: Bandwidth=0.1. Nadaraya-Watson estimator is used. . . . 30

3.3 DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: IV smile with strike, right panel: SPD. Bandwidth $h = 700$, the second order local polynomials is used. . . . 31

3.4 DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: IV smile with strike, right panel: SPD. bandwidth $h = 0.4$, the second order local polynomials is used. . . . 32

5.1 IV-Strike, DAX call option on 02-25-03, tau=0.14167, Top: IV smile. Bottom: SPD, red horizon line represents 0. Left: 4th order b-spline, right: 6th order B-spline . . . 44

5.2 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^7$, The blue dashed horizontal line represents 0. . . . 45

5.3 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^8$, The blue dashed horizontal line represents 0. . . . 46

5.4 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^9$, The blue dashed horizontal line represents 0. 47

5.5 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^14$, The blue dashed horizontal line represents 0. 48

5.6 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^16$, The blue dashed horizontal line represents 0. 49

5.7 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 5 \times 10^20$, The blue dashed horizontal line represents 0. 50

5.8 GCV score with respect to smoothing parameter λ, left panel: $L = D^2$, right panel: $L = D^3$. 51

5.9 DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with strike. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative of volatility with respect to strike. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line). 53

5.10 DAX call option, 02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with strike. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). 54

5.11 CV score with respect to smoothing parameter λ, left panel: $L = D^2$, right panel: $L = D^3$. 55

5.12 DAX call option, $L = D^2$. Upper left panel: IV smile. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative of volatility. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line). 56

5.13 DAX call option, $L = D^3$. Upper left panel: IV smile. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivatives. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line). 57

5.14 DAX call option,02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). 59

5.15 DAX call option,02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). 60

5.16 DAX call option,02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). 62

5.17 DAX call option,02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). 63

List of Tables

2.1 DAX option on 25th Feb, 2003 15

2.2 IV of DAX option on 022503 17

4.1 Examples of differential operators and the bases for the corresponding parametric families. 38

5.1 GCV score comparison 51

5.2 Parameters, Closing Day Data 56

5.3 Parameters 58

5.4 Parameters 62

1 Introduction

A derivative (derivative security or contingent claim) is a financial instrument whose value depends on the value of others, more basic underlying variables, see Franke, Härdle and Hafner (2003). In modern financial theory, the pricing of different financial derivatives based on these underlying varibales is always the most focal topic. The research of Black and Scholes (1973) sets up a milestone for this topic. Their well-known contribution, Black-Scholes option pricing model, gives the framework for the pricing of European style financial derivatives based on a set of assumptions. In B-S formula, the price of a financial derivative is determined by six parameters: the current underlying asset price S_t, the strike price K, the time to maturity τ, the riskless interest rate r, the dividend yield δ, and a constant volatility σ.

$$C^{BS}(S_t, t, K, T, \sigma, r, \delta) = e^{-\delta\tau} S_t \Phi(d_1) - e^{-r\tau} K \Phi(d_2) \tag{1.1}$$

$$d_1 = \frac{\ln(\frac{S_t}{K}) + (r - \delta + \frac{1}{2}\sigma^2)\tau}{\sigma\sqrt{\tau}}$$
$$d_2 = d_1 - \sigma\sqrt{\tau}$$

$\Phi(d)$ is the cumulative standard normal density function with upper integral limit d.

The first five parameters S_t, K, τ, r, δ can be observed from the financial market directly, while volatility can not. It can be recovered from B-S formula given other five parameters and option price. The most straightforward method suggested is to estimate volatility from financial market data, and insert this empirical volatility into (1.1) to calculate option price. However, the study of different financial markets (SP 500, FTSE, DAX and others) yields almost the same result: volatility is not constant. It displays a functional structure with respect to strike and time to maturity. As shown in figure 1.1, this is a contradiction to the constant volatility assumption of Black-Scholes formula, and volatility is rather a implicit function dependent on both strike and maturity. `Implied volatility` is defined as the parameter σ that when inserted into the B-S formula can actually yield the observed price of a particular option. That is "makes the B-S formula fit market prices of options". If we plot the implied volatility with respect to strikes and time to maturity, it exhibits a surface like a smile across strikes and time to maturity. This surface is so called `implied volatility surface`.

A large variety of applications on implied volatility have been introduced and practiced in financial market: it is an essential argument in smile consistent option pricing, it serves as a necessary tool in risk management, it is incorporated into market models for portfolio hedging, and volatility trading is a common practice on trading floor. At the same time, the idea of IV enlightens a lot of related researches which have extended the basic BS theory to a much wider range in many aspects.

The implied volatilities attained from financial market are contaminated by noise(caused by the microstructure of financial market), so we need to use some smoothing techniques to construct the IV smile. Quite a few researches have been made on modeling IV smile. The modelling can be classified into two large groups by different methods: parametric methods and non- and semi-parametric methods. Shimko (1993), Tompkins (1999) and Ané and German (1999) are among the first group. Non- and semi-parametric methods are thought to be a superior candidate than parametric methods since they allow high functional flexibility and parsimonious modeling. In an attempt to allow more flexibility, Hafner and Wallmeier (2001) fit quadratic splines to the smile function. Aït-Sahalia and Luo (1998), Rosenberg (2000), Cont, Fonseca and Durrleman (2002) and Fengler (2004) employed a Nadaraya-Watson estimator as the IVS, and higher order local polynomial smoothing of the IVS is used in Rookey (1997). Fengler (2004) introduced a least squares kernel estimator to smooth the IVS in the space of option prices.

The aim of this paper is to estimate IV smile with techniques known as functional data analysis. The technique that we will apply to the estimation is the so called "penalized B-splines". This method employs a `linear differential operator` as roughness penalty operator, which allows more flexibility in smoothing function. We will explain this method in more detail in later chapters. As the estimation doesn't assume that the IV function is arbitrage free, the smoothed IV smile will be inserted into B-S model and compute the `State Price Density(SPD)`. We will also compare the B-splines with other smoothing techniques such as Nadaraya-Watson or higher order local polynomial estimators.

In chapter 2, we will introduce some general idea that will be employed in the following chapters. First, we will give some statistical property of IV. Next, we will introduce the method of calculating IV from financial market data. Then, we will explain the concept of SPD and how to estimate it from IV smile. In the last section of this chapter, we will discuss some arbitrage free constraint on IV smile estimation.

The Nadaraya-Watson and higher order local polynomial estimators will be presented and applied to IV and SPD estimation in chapter 3 .

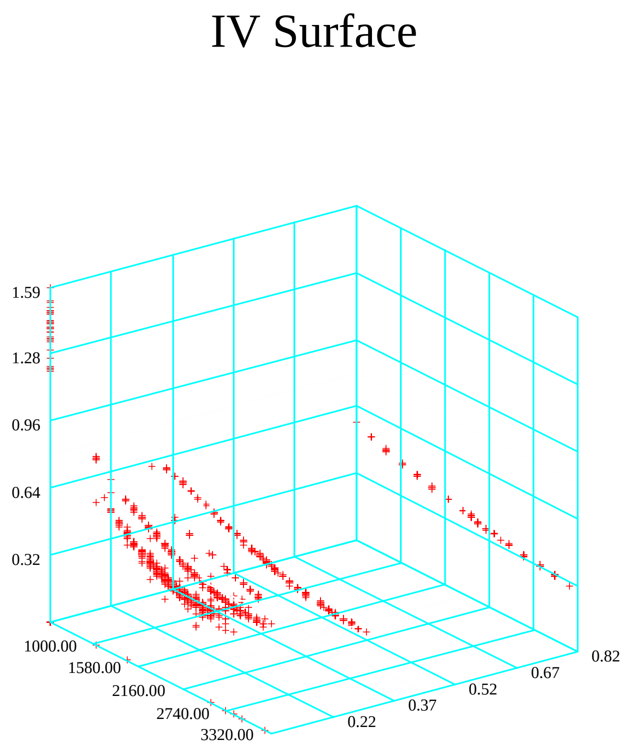


Figure 1.1: DAX option on 2003-02-25, Implied volatility with respect to strike and maturity `iv3d.xpl`

Chapter 4 is devoted to the penalized B-splines technique. We will define the functional data in the first section, next the definition of B-splines will be interpreted. The roughness penalty approach will be introduced in the last section of this chapter.

In chapter 5 we will apply different smoothing techniques to the data from financial market. We will implement penalized B-splines on the same data set. First, the penalty operator and smoothing parameters will be changed to see their impact on the estimation. To overcome the problem of multiple observations, the data will be preprocessed in the next section. The chapter concludes with a summary of the comparison of different smoothing techniques.

Chapter 6 summarizes and gives an outlook for ongoing work. An appendix contains the XploRe quantlets and the proofs.

2 The Implied Volatility and SPD

2.1 Statistical Properties of Implied Volatility

Quite a few empirical researches have been worked on Implied Volatility, they mainly focus on three following aspects: profile across strike level (smile patters), profile across maturity (term structure) and time series behavior of implied volatility. The studies on the behavior of implied volatility of traded option by using different market indices (SP 500, FTSE, DAX and others) give some statistical properties that seem to be common to these markets. Most striking feature of IV is that at a given date, the IV surface has a non-flat shape and exhibits both strike and term structure.

We use the dataset of DAX options on 25th February 2003 to illustrate the IV changes with strike and maturity, table 2.1 is a sample of the data:

Asset price	strike	Interest rate	maturity	option price	option type
2464.69	1200.00	0.02654	0.14167	1.00	0
2464.19	1200.00	0.02654	0.14167	1.00	0
2468.18	1200.00	0.02654	0.14167	1.10	0
2469.68	1200.00	0.02654	0.14167	1.10	0
2472.18	1200.00	0.02654	0.14167	1.10	0
2473.18	1200.00	0.02654	0.14167	1.10	0
2466.69	1400.00	0.02654	0.14167	3.00	0
2471.18	1600.00	0.02654	0.14167	7.00	0
2472.18	1600.00	0.02654	0.14167	7.00	0
2484.16	1600.00	0.02654	0.14167	6.80	0

Table 2.1: DAX option on 25th Feb, 2003

For convenience, we employ time to maturity $\tau \overset{def}{=} T - t$, and the so called future moneyness: $\kappa_f \overset{def}{=} K/F_t$, where $F_t = e^{(r-\delta)\tau}S_t$, is the forward price of stock at time t. We call the options with $\kappa = 1$ At The Money(ATM) option. Options close to ATM are usually traded with high frequency. The ATM options are most liquid and therefore are most available for empirical research.

Fengler (2004) concludes some static stylized facts of IVS observed on different capital market.

(1) The smile is very pronounced for short time to maturity and becomes more and more shallow for longer time to maturities.

(2) The smile function achieves its minimum in the neighborhood of ATM to near OTM call ($\kappa > 1$) option.

(3) OTM put ($\kappa < 1$) regions display higher levels of IV than OTM call region.

(4) The volatility of IV is biggest for short maturity options and monotonically decline with time to maturity.

2.2 Calculation of the Implied Volatility

Regardless of the invalid constant volatility assumption of the Black-Scholes model, in practice, the implied volatility can be obtained by inverting the Black-Scholes formula given the market price of the option.

In light of Black-Scholes European call option formula 1.1, we can get

$$\hat{\sigma} : C^{BS}(S_t, t, K, T, \hat{\sigma}) - \tilde{C}_t = 0$$

As we have already observed the evidently curvature shape of the IV σ across the strike K and not so evidently across maturity, it can be described as a function of time, strike prices and expiry days to R^+, and if the fixed data t is given, we have :

$$\hat{\sigma} : (t, K, T) \rightarrow \ \hat{\sigma}_t(K, T)$$

And this function of the IV σ is called implied volatility surface.

Using moneyness $\kappa_f \overset{def}{=} K/F_t$, the function of IV σ on given date t can be transformed as:

$$\hat{\sigma}_t(K, T) \rightarrow \ \hat{\sigma}_t(\kappa, \tau)$$

We use a European call as the example to get the volatility σ. and the implied volatility of a European put on same underlying with same strike and maturity can be deduced from the "put-call parity":

$$C_t - P_t = S_t - Ke^{-r\tau}$$

The implied Black-Scholes volatility can be determined uniquely from traded option prices because of the monotonicity of the Black-Scholes formula in the volatility parameter:

$$\frac{\partial S_t}{\partial \sigma_t} > 0$$

The Newton-Raphson algorithm provides a numerical way to invert the Black-Scholes formula in order to recover σ from the market prices of the option. XploRe provides a quantlet for this calculation (Härdle, Kleinow and Stahl (2002)), the usage of XploRe see Härdle, Klinke and Müller (2000)

```
y = ImplVola(x {, IVmethod})
```
calculates the implied volatility using either the method of bisections or the default Newton-Raphson method, the default method is Newton-Raphson method

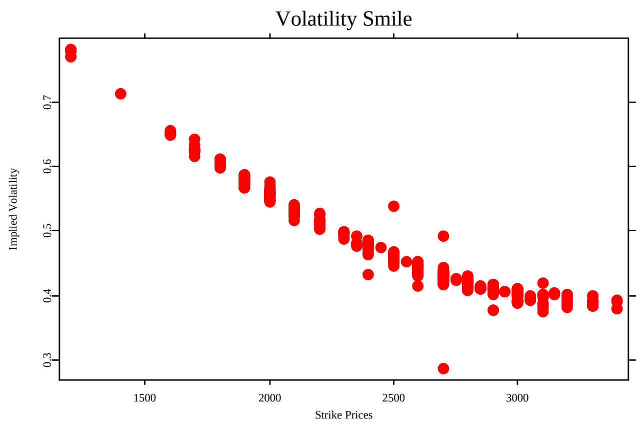
Table 2.2 provides the IV calculated by the quantlet. The data is from DAX options in 25th February 2003. Figure 2.1 plots volatility with strike and moneyness respectively.

asset price	strike	interst rate	maturity	option price	option type	IV
2464.69	1200.00	0.02654	0.14167	1.00	0	0.78135
2464.19	1200.00	0.02654	0.14167	1.00	0	0.77006
2468.18	1200.00	0.02654	0.14167	1.10	0	0.77988
2469.68	1200.00	0.02654	0.14167	1.10	0	0.78171
2472.18	1200.00	0.02654	0.14167	1.10	0	0.78043
2473.18	1200.00	0.02654	0.14167	1.10	0	0.76987
2466.69	1400.00	0.02654	0.14167	3.00	0	0.71348
2471.18	1600.00	0.02654	0.14167	7.00	0	0.64949
2472.18	1600.00	0.02654	0.14167	7.00	0	0.65397
2484.16	1600.00	0.02654	0.14167	6.80	0	0.65165

Table 2.2: IV of DAX option on 022503 `ivsmile.xpl`

2.3 State Price Density

State price density(SPD) is a fundamental concept in arbitrage pricing theory. The state price density can be understood as the probability density function of underlying



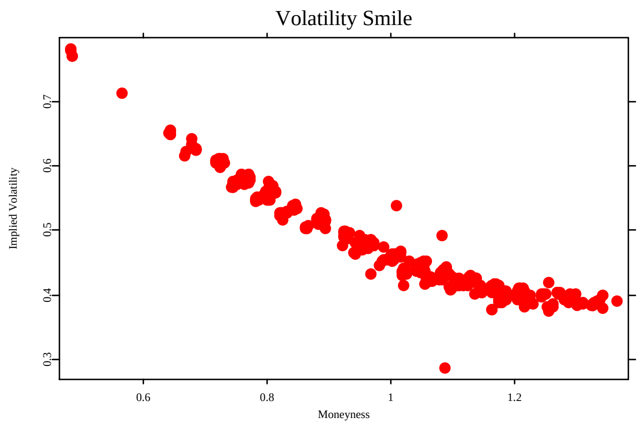


Figure 2.1: DAX option on 25th Feb, 2003, $\tau = 0.14167$. Top panel: IV VS strike; bottom: IV vs moneyness

ivsmile.xpl

asset price, under which the current price of each security is equal to the present value of the discounted expected value of its future payoffs given a risk-free interest rate.

Recall that the Black-Scholes European call option formula is:

$$C^{BS}(S_t, t, K, T, \sigma, r, \delta) = e^{-\delta\tau} S_t \Phi(d_1) - e^{-r\tau} K \Phi(d_2), \tag{2.1}$$

where S_t is the price of underlying asset at time t, δ is the expected dividends paid over the option's life, K is the option's strike price, τ is the time to maturity, r is the risk-free interest rate,

$$d_1 = \frac{\ln(\frac{S_t}{K}) + (r - \delta + \frac{1}{2}\sigma^2)\tau}{\sigma\sqrt{\tau}}$$

$$d_2 = d_1 - \sigma\sqrt{\tau}$$

The price of a european call is the present value of the discounted expected value of its future payoffs given the pdf of underlying asset price $f(S_t)$. It can be described as:

$$C_t(K,T) = e^{-r(\tau)} \int_0^\infty (S_t - K)_+ f(S_t) d(S_t) \tag{2.2}$$

From the definition of SPD, the pdf of underlying asset price $f(S_t)$ is just SPD. Redefine it as $\phi(K,T|S_t,t)$, deduced from 2.2, SPD is the discounted second derivative of call option price with respect to the strike price:

$$\phi(K,T|S_t,t) = e^{r(\tau)} \frac{\partial^2 C_t(K,t)}{\partial K^2} \tag{2.3}$$

The probability that the stock arrived at level $K \in [K_1, K_2]$ at date T, given that the stock is at level S_t in t, is computed by:

$$Q(S_T \in [K_1, K_2]) = \int_{K_1}^{K_2} \phi(K,T|S_t,t) dK. \tag{2.4}$$

The formula of the second derivative of call option price with respect to the strike price is:

$$\frac{\partial^2 C_t(K,t)}{\partial K^2} = \frac{e^{-r(\tau)}\varphi(d_2)}{\sigma\sqrt{\tau}K} = \frac{e^{-\delta(\tau)} S_t \varphi(d_1)}{\sigma\sqrt{\tau}K^2} \tag{2.5}$$

which yields the specific B-S transition density as:

$$\phi(K,T|S_t,t) = \frac{\varphi(d_2)}{\sigma\sqrt{\tau}K} \tag{2.6}$$

which is a log-normal pdf in K. It is proved that this log-normal pdf can be written as:

$$\ln S_T \sim N(\ln S_t + (r - \delta - \frac{1}{2}\sigma^2)\tau, \sigma^2\tau) \tag{2.7}$$

Where σ is a constant. The proof can be seen in Fengler (2004).

2.4 SPD estimation based on IV smile

The motivation for calculating the SPD given IV smile is to insert the IV as a function of strike or moneyness into the B-S call option price formula, and calculate the second derivative of call oprion price with respect to the strike price. Two different method will be introduced here, one is to estimate IV as a function of strike, the other is calculated in moneyness.

Recall that the SPD is the discounted second derivative of call oprion price with respect to the strike price:

$$\phi(K,T|S_t,t) = e^{r(\tau)}\frac{\partial^2 C_t(K,t)}{\partial K^2} \tag{2.8}$$

When strike is an explicit function of volatility, the approach to calculate the SPD is discussed in Drescher (2003). It can be derived as below.

First, we need to prove that following relationship always holds:

$$F_t\varphi(d_1) = K\varphi(d_2) \tag{2.9}$$

The right hand side can be presented in an alternative way:

$$\begin{aligned} K\varphi(d_2) &= K\varphi(d_1 - \sigma\sqrt{\tau}) \\ &= \frac{K}{\sqrt{2\pi}}\exp[-\frac{(d_1-\sigma\sqrt{\tau})^2}{2}] \\ &= \frac{K}{\sqrt{2\pi}}\exp[-\frac{(d_1^2 - 2d_1\sigma\sqrt{\tau}+\sigma^2\tau)}{2}] \\ &= \frac{K}{\sqrt{2\pi}}\exp[-\frac{d_1^2}{2}]\exp[d_1\sigma\sqrt{\tau}-\frac{\sigma^2\tau}{2}] \\ &= K\varphi(d_1)\exp[d_1\sigma\sqrt{\tau}-\frac{\sigma^2\tau}{2}] \end{aligned} \tag{2.10}$$

The exponential part of the right hand side is given as:

$$\begin{aligned} d_1\sigma\sqrt{\tau} - \frac{\sigma^2\tau}{2} &= \frac{\ln\frac{S_t}{K} + (r+\frac{\sigma^2}{2})\tau}{\sigma\sqrt{\tau}}\sigma\sqrt{\tau} - \frac{\sigma^2\tau}{2} \\ &= \ln\frac{S_t}{K} + (r+\frac{\sigma^2}{2})\tau - \frac{\sigma^2\tau}{2} \\ &= \ln\frac{S_t}{K} + r\tau \end{aligned} \tag{2.11}$$

Substitute this equation in to the exponent and get:

$$\begin{aligned} K\varphi(d_2) &= K\varphi(d_1)\exp[\ln\frac{S_t}{K}+r\tau] \\ &= S_t e^{r\tau}\varphi(d_1) \\ &= F_t\varphi(d_1) \end{aligned}$$

Based on this equation, the Black-Scholes formula for European call option can be rewritten as:

$$C_t(K,t) = e^{-r\tau}[F_t\Phi(d_1) - K\Phi(d_1-\sigma\sqrt{\tau})] \tag{2.12}$$

Under the assumption that the volatility is a function of K, the first derivative of the call option price with respect to strike is:

$$\begin{aligned} \frac{\partial C_t(K,t)}{\partial K} &= e^{-r\tau}[F_t\varphi(d_1)\frac{\partial d_1}{\partial K} - \Phi(d_1-\sigma\sqrt{\tau}) - K\varphi(d_1-\sigma\sqrt{\tau})(\frac{\partial d_1}{\partial K}-\frac{\partial\sigma}{\partial K}\sqrt{\tau})] \\ &= e^{-r\tau}[F_t\varphi(d_1)\frac{\partial d_1}{\partial K} - \Phi(d_2) - K\varphi(d_2)\frac{\partial d_1}{\partial K} + K\varphi(d_2)\frac{\partial\sigma}{\partial K}\sqrt{\tau}] \\ &= e^{-r\tau}[\frac{\partial d_1}{\partial K}\{F_t\varphi(d_1)-K\varphi(d_2)\} - \Phi(d_2) + K\varphi(d_2)\frac{\partial\sigma}{\partial K}\sqrt{\tau}] \\ &= e^{-r\tau}[-\Phi(d_2) + K\varphi(d_2)\frac{\partial\sigma}{\partial K}\sqrt{\tau}] \end{aligned} \tag{2.13}$$

Differentiate the equation 2.13 again with respecct to strike K:

$$\begin{aligned} \frac{\partial^2 C_t(K,t)}{\partial K^2}e^{r\tau} &= -\varphi(d_2)\frac{\partial d_2}{\partial K} + \sqrt{\tau}[\{\varphi(d_2)+K\frac{\partial\varphi(d_2)}{\partial d_2}\frac{\partial d_2}{\partial K}\}\frac{\partial\sigma}{\partial K} + K\varphi(d_2)\frac{\partial^2\sigma}{\partial K^2}] \\ &= \varphi(d_2)[-\frac{\partial d_2}{\partial K} + \sqrt{\tau}\frac{\partial\sigma}{\partial K} - \sqrt{\tau}Kd_2\frac{\partial\sigma}{\partial K}\frac{\partial d_2}{\partial K} + \sqrt{\tau}K\frac{\partial^2\sigma}{\partial K^2}] \end{aligned} \tag{2.14}$$

where followng relationship is used:

$$\begin{aligned} \frac{\partial\varphi(d_2)}{\partial d_2} &= \frac{\partial}{\partial d_2}(\frac{1}{\sqrt{2\pi}}\exp[-\frac{d_2^2}{2}]) \\ &= -d_2\varphi(d_2) \end{aligned} \tag{2.15}$$

And further $\frac{\partial d_2}{\partial K}$ can be substituted as:

$$\begin{aligned} \frac{\partial d_2}{\partial K} &= \frac{\partial}{\partial K}(\frac{\ln S_t - \ln K + r\tau - \frac{\sigma^2\tau}{2}}{\sigma\sqrt{\tau}}] \\ &= \frac{\sigma\sqrt{\tau}[-\frac{1}{K}-\tau\sigma\frac{\partial\sigma}{\partial K}] - \sqrt{\tau}\frac{\partial\sigma}{\partial K}[\ln S_t - \ln K + r\tau - \frac{\sigma^2\tau}{2}]}{\sigma^2\tau} \\ &= -\frac{1}{K\sigma\sqrt{\tau}} - \sqrt{\tau}\frac{\partial\sigma}{\partial K} - \frac{d_2}{\sigma}\frac{\sigma}{K} \end{aligned} \tag{2.16}$$

Reorder and combine the above terms together we can achieve the result:

$$\frac{\partial^2 C_t(K,t)}{\partial K^2} = e^{-r\tau}\varphi(d_2)[\frac{1}{K\sigma\sqrt{\tau}} + (\frac{2d_1}{\sigma})\frac{\partial\sigma}{\partial K} + (\frac{d_1 d_2 K\sqrt{\tau}}{\sigma})(\frac{\partial\sigma}{\partial K})^2 + (K\sqrt{\tau})\frac{\partial^2\sigma}{\partial K^2}] \quad (2.17)$$

with:

$$\sigma = \sigma(K)$$

SPD can be calculated with:

$$\phi(K,T|S_t,t) = \varphi(d_2)[\frac{1}{K\sigma\sqrt{\tau}} + (\frac{2d_1}{\sigma})\frac{\partial\sigma}{\partial K} + (\frac{d_1 d_2 K\sqrt{\tau}}{\sigma})(\frac{\partial\sigma}{\partial K})^2 + (K\sqrt{\tau})\frac{\partial^2\sigma}{\partial K^2}] \quad (2.18)$$

From equation 2.18 we can get the spd equation with the measure of monyness. We always have $F_t = e^{(r-\delta)(\tau)}S_t, \kappa_f = K/F_t$

$$\frac{\partial\sigma(K)}{\partial K} = \frac{\partial\sigma(\kappa_f)}{\kappa_f}\frac{1}{F_t} \quad (2.19)$$

$$\frac{\partial^2\sigma(K)}{\partial K^2} = \frac{\partial^2\sigma(\kappa_f)}{\kappa_f^2}\frac{1}{F_t^2} \quad (2.20)$$

Insert 2.19 and 2.20 into 2.18, SPD can be computed with:

$$\begin{aligned}\phi(\kappa_f,T|S_t,t) &= \varphi(d_2)[\frac{1}{\kappa_f F_t\sigma\sqrt{\tau}} + (\frac{2d_1}{\sigma})\frac{\partial\sigma}{\partial\kappa_f}\frac{1}{F_t} + (\frac{d_1 d_2\kappa_f F_t\sqrt{\tau}}{\sigma})(\frac{\partial\sigma}{\partial\kappa_f})^2\frac{1}{F_t^2} \\ &+ (\kappa_f F_t\sqrt{\tau})\frac{\partial^2\sigma}{\partial\kappa_f^2}\frac{1}{F_t^2}] \quad (2.21)\end{aligned}$$

with:

$$\sigma = \sigma(\kappa_f)$$

The accompanied quantlet `spdcal` and `spdcalm` is introduced in the next subsection.

2.5 The Quantlet spdcal and spdcalm

```
fstar= spdcal(k, sig, sig1, sig2, s, r, tau)
```
Calculate the empirical state-price density

The quantlet `spdcal` uses function 2.18 to compute the SPD. The analytic formula uses an estimate of the volatility smile and its first and second derivative to calculate the State price density, This method can only be applied to European options (due to the assumptions).

The quantlet `spdcal` has the following input parameters.

`k`- $N \times 1$ vector of strike prices

`sig`- $N \times 1$ vector of points of the estimated volatility

`sig1 -` $N \times 1$ vector of points of the first derivative of the volatility smile (with respect to strike price)

`sig2 -` $N \times 1$ vector of points of the second derivative of the volatility smile (with respect to strike price)

`s -` $N \times 1$vector, underlying spot price corrected for dividends

`r -` $N \times 1$ vector, risk-free interest rate

`tau -` $N \times 1$ vector, time to maturity

Output Parameter

`fstar`- $N \times 1$ vector of state-price density

```
fstar= spdcalm(m, sig, sig1, sig2, s, r, tau)
```
Calculate the empirical state-price density with moneyness measure

The quantlet uses function 2.21 to compute the SPD. The analytic formula uses an estimate of the volatility smile with respect to moneyness and its first two derivatives to calculate the SPD, This method can only be applied to European options (due to the assumptions).

The quantlet `spdcal` has the following input parameters.

`m`- $N \times 1$ vector of moneyness

`sig`- $N \times 1$ vector of points of the estimated volatility

`sig1` - $N \times 1$ vector of points of the first derivative of the volatility smile (with respect to strike price)

`sig2` - $N \times 1$ vector of points of the second derivative of the volatility smile (with respect to strike price)

`s` - $N \times 1$vector, underlying spot price corrected for dividends

`r` - $N \times 1$ vector, risk-free interest rate

`tau` - $N \times 1$ vector, time to maturity

Output Parameter

`fstar`- $N \times 1$ vector of state-price density

2.6 Arbitrage Free Constraint

A lot of recent researches have contributed to estimation of no-arbitrage SPD. The practical way to estimate SPD might be roughly classified into 2 groups. One is the direct approach based on the observation of option price, to fit the observations of option price to the theoretical option prices, and derive the prespecified SPD from the second derivatives of option price with respect to strike. Härdle and Hlávka (2005) have used non parametric method to estimate SPD from DAX option price data with this direct approach. The other is the indirect approach based on volatility smile. First recover volatility smile into the call price and then get SPD. This indirect method have been discussed in section 2.4, it was first introduced by Shimko (1993).

No matter what method has been employed, the difficulty of estimation is the imposition of arbitrage-free constraint. The call price function, according to Härdle and Hlávka (2005), has following no-arbitrage constraint:

1. it is positive
2. it is decreasing in K
3. it is convex
4. its second derivative exists and it is a density (i.e.,nonnegative and it integrates to one)

The constraint 1 and 2 is obvious, the constraint 4 is derived from general property of density function. The constraint 3 is deduced from arbitrage free condition requirements, see 7.2.1.

The shape constraint of European call price can be "translated" to the shape constraint of volatility smile. But this translation process is easy to explain in theory but bitter in practice. Below we list some general shape constraint for volatility smile through the strike direction (not restricted to no-arbitrage constraint).

1. Estimated volatility should be positive.

$$\hat{\sigma} \geq 0 \tag{2.22}$$

2. There exists the lower bound and upper bound of the first derivative of volatility with respect to strike K. This can be expressed as:

$$-\frac{\Phi(-d_1)}{\sqrt{\tau}K\varphi(d_1)} \leq \frac{\partial\hat{\sigma}}{\partial\kappa_f} \leq \frac{\Phi(-d_2)}{\sqrt{\tau}K\varphi(d_2)} \tag{2.23}$$

   Using $F_t = e^{(r-\delta)(\tau)}S_t, \kappa_f = K/F_t$, this can be expressed in terms of moneyness measure:

$$-\frac{\Phi(-d_1)}{\sqrt{\tau}\kappa_f\varphi(d_1)} \leq \frac{\partial\hat{\sigma}}{\partial\kappa_f} \leq \frac{\Phi(-d_2)}{\sqrt{\tau}\kappa_f\varphi(d_2)} \tag{2.24}$$

3. the right hand side of 2.18 is nonnegative and integrates to one, we can obtain

$$\frac{1}{K\sigma\sqrt{\tau}} + \left(\frac{2d_1}{\sigma}\right)\frac{\partial\sigma}{\partial K} + (\frac{d_1 d_2 K\sqrt{\tau}}{\sigma})(\frac{\partial\sigma}{\partial K})^2 + (K\sqrt{\tau})\frac{\partial^2\sigma}{\partial K^2} \geq 0 \tag{2.25}$$

$$\int_K \Phi(K,T|S_t,t)dK = 1 \tag{2.26}$$

   When K is on the limit interval $[K_l, K_h]$

$$\int_{K_l}^{K_h} \Phi(K,T|S_t,t)dK \leq 1 \tag{2.27}$$

Constraint 2 can be proved, see Appendix 7.2.2. Constraint 3 is translated from the constraint 4 of call price function.

3 Nadaraya-Watson and Local Polynomial Estimator

There are several studies fitted parametric volatility functions to observed implied volatilities. Shimko (1993), Tompkins (1999), Ané and German (1999) and Ané and German (1999) are among this group. They usually employ quadratic specifications to model the IV function along the strike profile. These parametric approaches are not able to capture the prominent feature of IVS patterns, and hence the estimates maybe biased. Others are focus on non-and semiparametric smoothing approaches to estimate IVS. Aït-Sahalia and Luo (1998), Rosenberg (2000), Cont et al. (2002) and Fengler (2004) applied a Nadaraya-Watson estimator to the IVS, and higher order local polynomial smoothing of the IVS is used in Rookey (1997). Below we'll give a short introduction to this non-parametric method.

For simplicity, consider the univariate model

$$Y = m(X) + \varepsilon \tag{3.1}$$

with the unknown regression function m. The explanatory variable X and the response variable Y take values in $\mathbb{R}$, have the joint pdf $f(x, y)$ and are independent of ε. The error ε has the properties $E(\varepsilon|x) = 0$ and $E(\varepsilon^2|x) = \sigma^2(x)$.

Taking the conditionalexpectation of 3.1 yields

$$E(Y|X = x) = m(x) \tag{3.2}$$

which says that the unknown regression function is the conditional expectation function of Y given $X = x$. Using the definition of the conditional expectation 3.2 can be written as:

$$m(x) = E(Y|X = x) = \frac{\int y f(x, y) dy}{f_x(x)}, \tag{3.3}$$

where $f_x(x)$ denotes the marginal pdf. Representation 3.3 shows that the regression function m can be estimated via the kernel density estimates of the joint and the marginal density.

In the book of Härdle, Müller, Sperlich and Werwatz (2004) the Nadaraya-Watson estimator is introduced:

$$\hat{m}(x) = \frac{n^{-1}\sum_{i=1}^{n} K_h(x-x_i)y_i}{n^{-1}\sum_{i=1}^{n} K_h(x-x_i)} \tag{3.4}$$

Rewriting above formula as

$$\hat{m}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{K_h(x-x_i)}{n^{-1}\sum_{j=1}^{n} K_h(x-x_j)} y_i = \frac{1}{n}\sum_{i=1}^{n} \omega_{i,n}(x) y_i \tag{3.5}$$

This formula indicates that the Nadaraya-Watson estimator can be seen as a weighted (local) average of the response variables with weights:

$$\omega_{i,n}(x) \overset{def}{=} \frac{K_h(x-x_i)}{n^{-1}\sum_{j=1}^{n} K_h(x-x_j)} \tag{3.6}$$

Here $K_h(\bullet)$ denotes univariate Kernel function, Quartic kernel or Gaussian kernel are the most widely used kernels:

Quartic(Biweight)kernel

$$K_u = \frac{15}{16}(1-u^2)^2 I(|u| \le 1) \tag{3.7}$$

Gaussian kernel

$$K_u = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}u^2) \tag{3.8}$$

The Nadaraya-Watson estimator can be seen as a special case of a larger family of Local smoothing estimators. One generalization is Local PolynomialRegression.

$$min_\beta \sum_{i=1}^{n} \{(Y_i - \beta_0 - \beta_1(X_i - x) - ... - \beta_p(X_i - x)^p)\}^2 K_h(x - X_i) \tag{3.9}$$

where β denotes the coefficients vector$\beta = (\beta_0, \beta_1, ..., \beta_p)^\top$

we can compute β by using weighted least squares estimator with weights $K_h(x_i - x)$, it has been studied extensively by Fan and Gijbels (1996).

We introduce the following matrix notation:

$$X = \begin{pmatrix} 1 & x-x_1 & (x-x_1)^2 & \dots & (x-x_1)^p \\ 1 & x-x_2 & (x-x_2)^2 & \dots & (x-x_2)^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x-x_n & (x-x_n)^2 & \dots & (x-x_n)^p \end{pmatrix}, \tag{3.10}$$

and $Y = (y_1, ..., y_n)^\top$, and the weight matrix:

$$X = \begin{pmatrix} K_h(x - x_1) & 0 & \dots & 0 \\ 0 & K_h(x - x_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & K_h(x - x_n) \end{pmatrix}, \tag{3.11}$$

Then we can write the solution of 3.9 in the usual LSE formulation as:

$$\hat{\beta}_x = (X^\top W X)^{-1} X^\top W Y \tag{3.12}$$

For $p = 0, \hat{\beta}$ reduces to β_0, we get Nadaraya-Watson estimator. And the order p is usually taken to be one (local linear) or three (local cubic regression). Nadaraya-Watson regression corresponds to a local constant least squares fit.

We plot IV smile by using Nadaraya-Watson regression figure 3.1, 3.2 and second order local polynomial regression figure 3.3, 3.4. In Nadaraya-Watson case, when the bandwidth is small, the estimator failed to generate a continuous function; if we enlarge the bandwidth, the large bias appears and the curve can't fit to the data, especially in the region where the observations are sparse. The local polynomial regression approach has at least 2 superiorities over Nadaraya-Watson estimator. The first is that local linear estimator improved the estimates in boundary regions. When we use Nadaraya-Watson approach, possibly some of the points are used more than ones to estimate the curve near the boundary. This problem is improved since Local polynomial imposes a higher degree polynomial here. The second is that if we use local polynomial approach, it will be easier to estimate the derivatives of function $m(\bullet)$. This advantage is very useful in estimating IV based state price density, since the estimation of SPD requires the estimation of IV and it's first two derivatives. We also plot the estimated SPD in figure 3.3, 3.4.

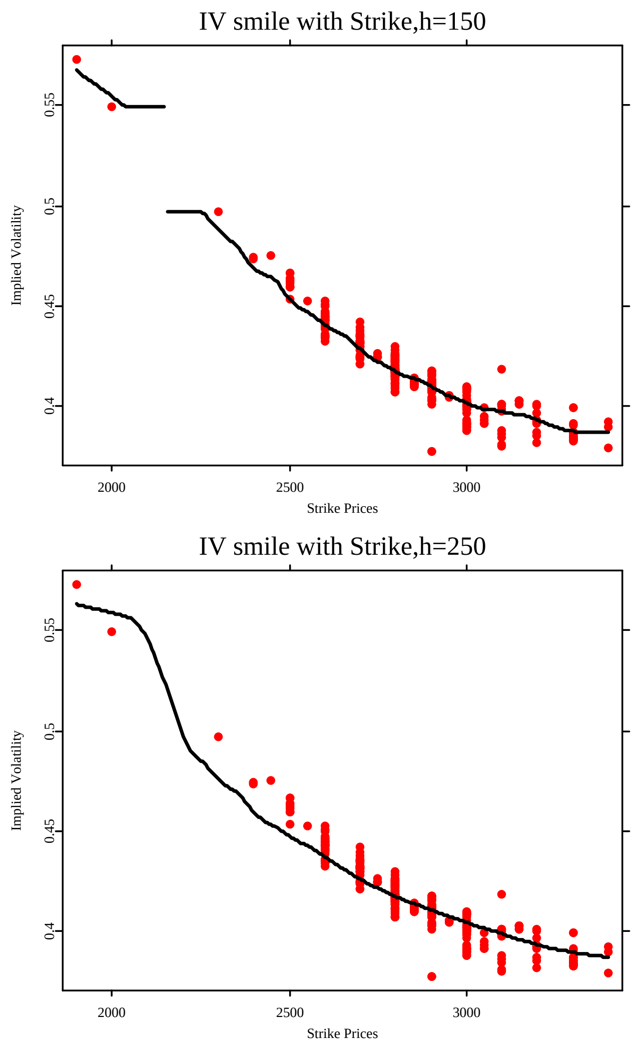


Figure 3.1: IV smile with strike, DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: Bandwidth=150, lower panel: Bandwidth=250. Nadaraya-Watson estimator is used. spdcalnw.xpl

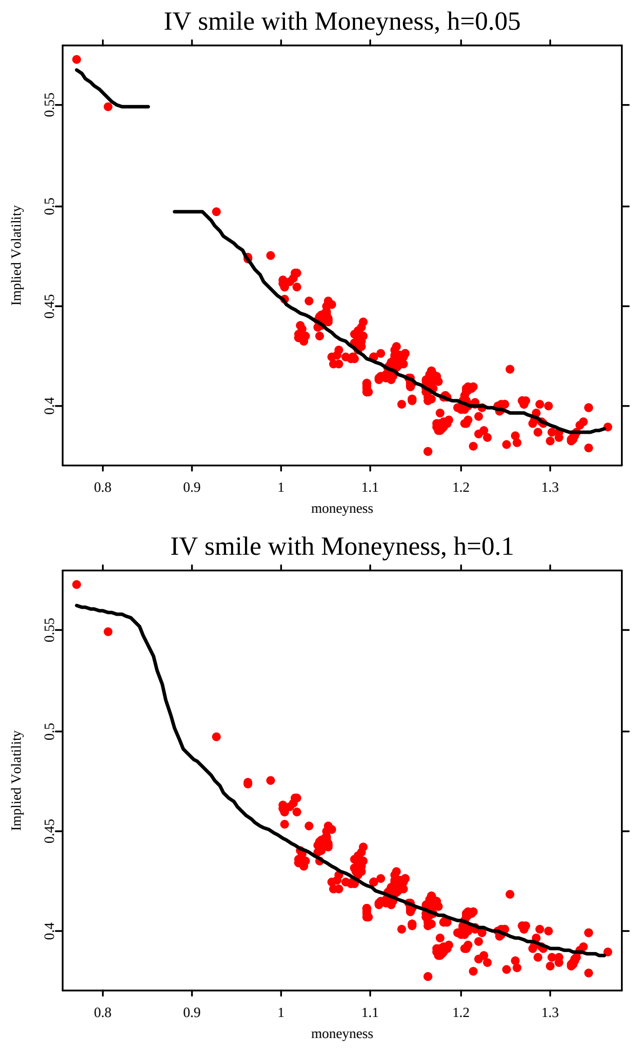


Figure 3.2: IV smile with moneyness, DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: Bandwidth=0.05, lower panel: Bandwidth=0.1. Nadaraya-Watson estimator is used. `spdcalnwm.xpl`

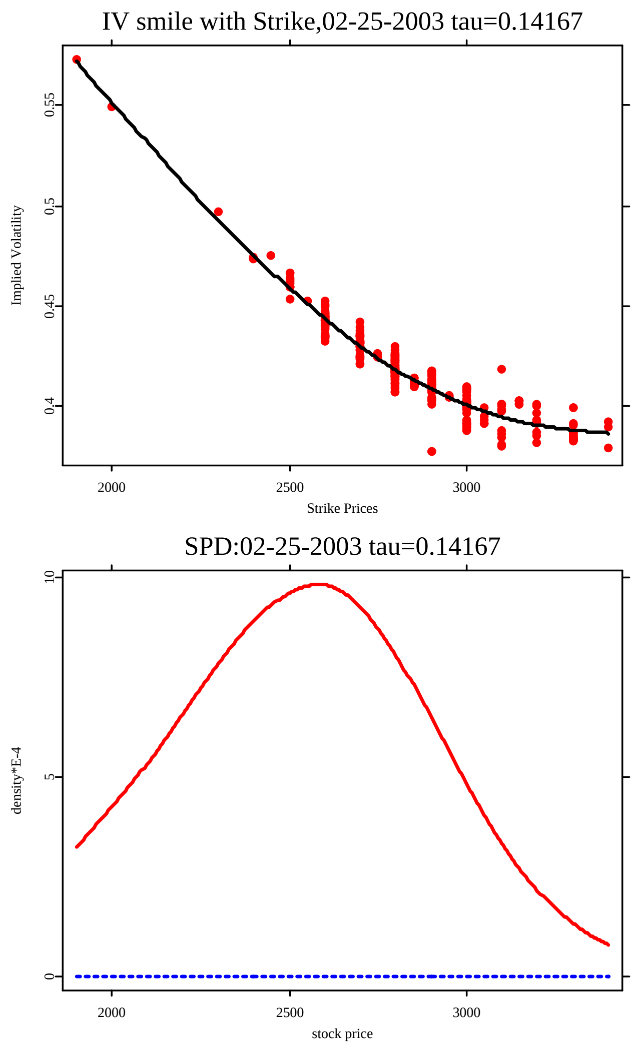


Figure 3.3: DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: IV smile with strike, right panel: SPD. Bandwidth $h = 700$, the second order local polynomials is used. spdcallp.xpl

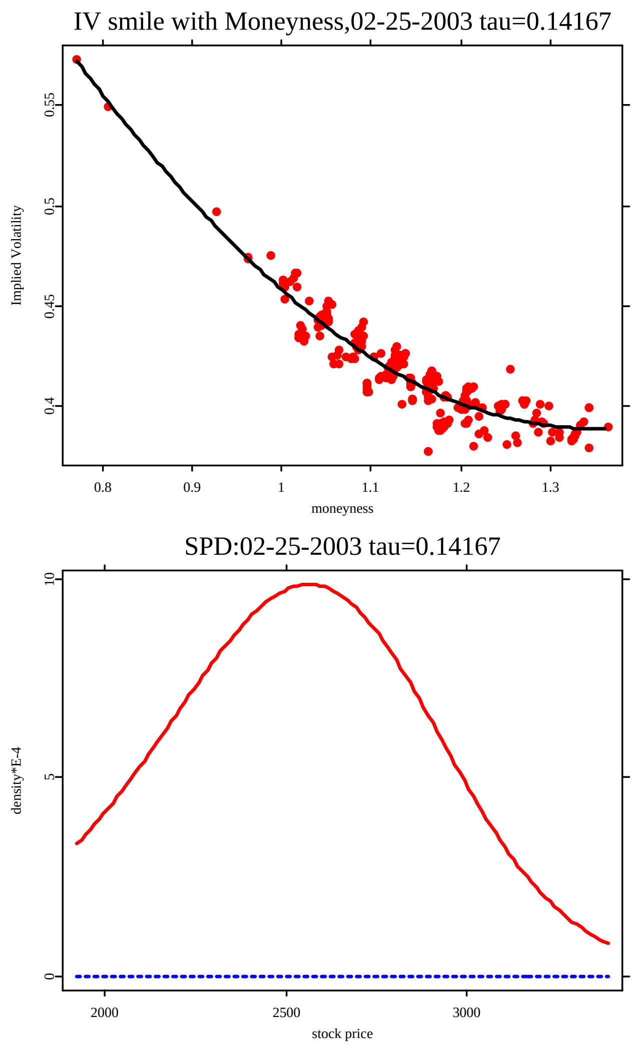


Figure 3.4: DAX call option,02-25-03, $\tau = 0.14167$. Upper panel: IV smile with strike, right panel: SPD. bandwidth $h = 0.4$, the second order local polynomials is used.
spdcallpm.xpl

4 Smoothing IV With Functional Data Approach

4.1 Basic Setup

In every working day, options are traded frequently in financial market. For each transaction, we can calculate one implied volatility. If we collect the data time by time and day by day, we can recover implied volatility to a function of strike K and maturity τ. Due to this attribute we may understand IV as functional data in that each observation is a real function of underlying variables. We can find some other examples of functional data in many fields, such as human growth, which is a function of time or age; weather data is another example since its dependence on other observations such as temperature and humidity. The basis function approach for representing functional data as smooth functions, provides us more flexible smoothing techniques for estimation out of discretely observed data.

Formally, a sample of multivariate data $\mathcal{X} = \{X_1, ..., X_n\}$ is described as $(n \times p)$ matrix containing n observations of p-dimensional row vectors of p (one dimensional) random variables $(2 \leq p < \infty)$. It is the measurable mapping $(\Omega, \mathcal{A}, \mathcal{P}) \rightarrow (\mathbb{R}^p, \mathcal{B})$, where $(\Omega, \mathcal{A}, \mathcal{P})$ is a probability space, Ω is the set of all elementary events, $\mathcal{A}$ is a sigma algebra of subsets of Ω, and $\mathcal{P}$ is a probability measure defined on $\mathcal{A}$. $\mathbb{R}^p$ is a p-dimensional vector space, $\mathcal{B}$ is a sigma algebra on $\mathbb{R}^p$.

In the case of functional data, $\mathcal{X}$ consists of random functions, we denote the ith observations as $X_i(t), i = 1, ..., n$ of some argument t. It is the measurable mapping $(\Omega, \mathcal{A}, \mathcal{P}) \rightarrow (\mathcal{H}, \mathcal{B}_{\mathcal{H}})$, where $\mathcal{H}$ is the space of functions.

We denote the i-th observation as $X_i(t), i = 1, ..., n$ of some argument t. It will be assumed that the functions are observed on a finite interval $J = [t_L, t_U] \in R$, where t_l and t_U denotes the lower and upper bound, respectively. Ramsay and Dalzell (1991) introduce the name functional data analysis (FDA) and give the following definition:

Functional data:

A sample $\mathcal{X} = \{X_1, ..., X_n\}$ is called functional data where the i-th observation is a real function $X_i(t), t \in J, i = 1, ..., n$, and hence, each $X_i(t)$ is a point in some function space $\mathcal{H}$.

A single functional observation, which means a single observed function, is called a replication. Functional data in turn is a random sample of replications. In this paper, we will use one-day data, fix the maturity and focus on estimating IV function of strike by smoothing techniques from functional data analysis. In our case, we have only one replication. More advanced method for functional data can be found in Ramsay and Silverman (1997)

4.2 The Basis Function Approach

We want to present functional data as smooth functions of some continuous parameters. Generally we have no idea in advance how these curves look like, how complex they will be and what certain characteristics they may have, typically we will assume just some degree of smoothness. For this reason flexibility is the key issue we demand for estimation. To solve this problem we need some mathematical "machinery", with which we can do any computation that is needed to fit the data quickly and flexibility.

The core idea is to replace the vector of inputs t with additional variables, which are transformations of t, and then use linear models in this new space of derived input features. See Hastie, Tibshirani and Jerome (2001).

For simplicity, consider the following univariate model:

$$Y = X(t) + \varepsilon \tag{4.1}$$

Our approach is to minimize the penalized residual sum of squares

$$min_{X \in \mathcal{H}_K} [\{Y - X(t)\}^\top \{Y - X(t)\} + \lambda PEN(X(t))] \tag{4.2}$$

where $\mathcal{H}_K$ is the Hilbert space of real function K. λ is a smoothing parameter, $PEN(X(t))$ is the penalty operator. We will discuss the penalized regression in more detail in Section 4.4.

According to Wahba (1990), the solution to 4.2 and even to a more general class of penalized regression problems has the form of finite-dimensional linear combination

$$\hat{X}(t) = \sum_{k=1}^{K} c_k \phi_k(t), c_k \in \mathbb{R}; K < \infty \tag{4.3}$$

where $\phi_k(t)$ is a real-valued function and following Hastie et al. (2001), it is referred to as basis function. This approach can be seen as a particular linear regression model with each variables being some basis functions. Once these functions $\phi_k(t)$ have been defined, we can fit them to classical linear model as before. And this approach in estimation $X(t)$ with 4.3 is called "Basis function Approach".

4.3 B-spline Bases

A wide range of basis function $\phi_(t)$ can be applied in this approach, the most popular are Fourier Bases, Polynomial bases and B-spline bases. B-spline will be introduced here since its flexibility in representing $X(t)$. If there exists a space of piecewise polynomials, B-spline is a basis of certain subspace of that space .

We divide the interval $J = [t_L, t_U]$ into several continuous intervals and represent $X(t)$ by separate polynomial in each interval. $X(t)$ is a piecewise polynomial function.

Follow the definition of de Boor (1978), the piecewise polynomial can be described as below:

Definition 1 (piecewise polynomial) *Divide the interval $J = [t_L, t_U]$ into m subintervals by a strictly increasing sequence of points $t_L = \xi_1 < \xi_2 < ... < \xi_m < \xi_{m+1} = t_U$. The points ξ_i are called breakpoints. Let K be a positive integer. Then the corresponding piecewise polynomial function $X(t)$ of order K is defined by*

$$X(t) \overset{def}{=} p(t) = \sum_{j=0}^{K-1} c_j t^j I\{x \in [\xi_i, \xi i + 1]\}, i = 1, ..., m \tag{4.4}$$

If we restrict that the piecewise function, some times their higher order derivatives to be continuous in each breakpoint, we will obtain the polynomial splines. de Boor (1978)also define the polynomial splines as below:

Definition 2(polynomial spline) *Let $K > 0$ be an integer indicating the order of piecewise polynomials, $\xi = \xi_{i}{}_{i=1}^{m+1}$ be a given breakpoint sequence with $t_L = \xi_1 < \xi_2 < ... < \xi_m < \xi_{m+1} = t_U$, the vector $\nu = \{\nu_i\}_{i=2}^{m}$ counts the number of continuity conditions at each interior breakpoint. If $\nu_i = K - 1$ for all $i = 2, ..., m$ then the piecewise polynomial function $X_i(t)$ is called polynomial spline of order K.*

B-splines has been defined as a divided difference of the truncated power basis. And it is formally defined as:

Definition 3(B-splines)

$$\theta_l(t) = B_{l,K}(t), l = 1, \dots, m + k - 2 \tag{4.5}$$

where $B_{l,K}$ is l-th B-Spline of order K, for the non-decreasing sequence of knots $\{\tau_i\}_{i=1}^{m}$

defined by following recursion scheme:

$$B_{i,1}(t) = \begin{cases} 1, & \text{for } t \in [\tau_i, \tau_{i+1}] \\ 0, & \text{otherwise} \end{cases}$$

$$B_{i,k}(t) = \frac{t-\tau_i}{\tau_{i+k-1}-\tau_i} B_{i,k-1}(t) + \frac{\tau_{i+k}-t}{\tau_{i+k}-\tau_{i+1}} B_{i+1,k-1}(t)$$

for $i = 1, \ldots, m+k,\ k = 0, \ldots, K.$

The number of the basis function will uniquely be defined by the B-spline order and the number of knots, while $nbasis = nknots + norder - 2$. Flexibility is one advantage of the B-spline basis and its also relatively easy to evaluate the basis functions and their derivatives. XploRe quantlet `Bsplineevalgd` can be used for this approach.

4.4 Estimation of Coefficients

Consider the basis function approach

$$f(x) = \sum_{k=0}^{K} c_k \phi_k(x) \tag{4.6}$$

Estimate c_k by the observed data $(x_j, y_j), j = 1, ..., n$

$$Y = \Phi C + \varepsilon \tag{4.7}$$

where $Y = (y_1, ...y_n)^\top, \{\Phi\}_k = \phi_k(x), C = (c_1, ..., c_K)^\top, \varepsilon = (\varepsilon_1, ...\varepsilon_n)^\top$.

The Generalized Least Squares (GLS) estimation of coefficient c is:

$$\hat{C} = (\Phi^\top \Phi)^{-1} \Phi^\top Y \tag{4.8}$$

Using 4.7 we leave the degree of smothness just by the order of B-spline. It is merely interpolating the data, without exploiting some structure noises might present in the data. We may use some additional information of function $f(x)$, and add a roughness penalty to 'penalize' the non smoothness.

Due to the insight of Ramsay (1997), it is the smoothness of the process generating functional data that differentiates this type of data from more classical multivariate observations. This smoothness ensures that the information in the derivatives of functions can be used in a reasonable way. We can make use of the fact that the curvature of the curve $f(x)$ increases with $||f''(x)||$, and get the total curvature as measure of roughness penalty

$$PEN_2(f) = \int_J \{D^2 f(s)\}^2 ds = ||D^2 f||^2 \tag{4.9}$$

When using basis expansion $f(X) = \Phi C$, the penalty term can be written as

$$PEN_2(f) = C^\top RC \tag{4.10}$$

where $\{R\}_{ij} = \int_J \{D^2\phi_i(s)\}\{D^2\phi_j(s)\}ds = \langle D^2\phi_i, D^2\phi_j\rangle$

Now we can match the observation with the smooth function by solving the minimization of penalized residual sum of squares function:

$$\hat{f}_\lambda = argmin_f PENSSE_\lambda(f|y) \tag{4.11}$$

$$PENSSE_\lambda(f|y) = (Y - \Phi C)^\top (Y - \Phi C) + \lambda C^\top RC \tag{4.12}$$

The parameter λ controls the weight given to the stabilizer in the minimization. The higher the λ is, the more weight is given to $||f''(x)||$, and the smoother is the estimator. λ can be chosen by Cross validation method.

The linear differential operator (LDO) can be used as a more generalized roughness Penalty.

$$Lf(x) = \omega_0(x) + \omega_1(x)Df(x) + ... + \omega_{m-1}D^{m-1}f(x) + D^m f(x) \tag{4.13}$$

The Generalized penalized residual sum of squares function:

$$PENSSE_\lambda(X|y) = (Y - \Phi C)^\top (Y - \Phi C) + \lambda ||Lf||^2 \tag{4.14}$$

where $||Lf||^2 = C^\top RC$, with $\{R\}_{ij} = \int_J \{D^m\phi_i(s)\}\{D^m\phi_j(s)\}ds = \langle D^m\phi_i, D^m\phi_j\rangle$

In practice, since many models are based on scientifical or phisical models, there is a general idea of the order and coefficients of LDO. Sometimes, we may be able to surmise the shape of smoothed curve, thus we can be motivated to choose special LDOs to incorporate this shape speculation. There are links between some linear differential operator and bases for the corresponding parameter families. Heckman and Ramsay (2000) provides table 4.1 for some examples of differential operators and the bases for the corresponding parametric families

4.5 Choice of the Smoothing Parameter

Technically, two possible approaches can be applied to choose the smoothing parameter λ. As discussed by Green and Silverman (1994). The first approach is to choose the smoothing parameter λ subjectively. With different λ, different features of the data that arise from different "scales" can be explored, and the one parameter value which "looks best" might be chosen. The second approach is to choose λ automatically

Operator L	Parametric family for the kernel of L
D^2	$\{1, t\}$
D^4	$\{1, t, t^2, t^3\}$
$D^2 + \gamma D, \gamma \neq 0$	$\{1, exp(-\gamma t)\}$
$D^4 + \omega^2 D^2$	$\{1, t, cos(\omega t), sin(\omega t)\}$
$(D^2 - \gamma D)(D^2 + \omega^2 D^2), \gamma, \omega \neq 0$	$\{1, exp(\gamma t), cos(\omega t), sin(\omega t)\}$
$D - \omega(\bullet)I, \omega(t) \neq 0$	$\{exp[\int \omega(u)du]\}$
$D^2 - \omega(\bullet)D, \omega(t) \neq 0$	$\{1, \int exp[\int \omega(v)dv]du\}$

Table 4.1: Examples of differential operators and the bases for the corresponding parametric families.

based on the data set its self, one popular method of which is cross validation. In many cases these two approaches can be combined, and an automatic choice can be used as a starting point for subsequent subjective adjustment.

Cross-validation seems to be the most frequently used methods for choosing the smoothing parameter. The main idea of this method is to split the data set into two parts, the "learning " sample and the "validation" sample. First, fit the data to the model by using learning sample then assess the fit to the validation sample, and finally choose the λ value by minimizing some error criterion. We can split the data by the " leave-one-out" rule. This method is to leave i-th observation out , and the function $X_i(t)$ is predicted from other functions in the data set.

Following Ramsay and Silverman (2002), let $m_\lambda^{-i}(t_i)$ denote the smoothed sample mean calculated with smoothing parameter λ from all observations except t_i. As a measure of global discrepancy compute the integrated squared error(ISE)

$$ISE\{m_\lambda^{-i}(t_i)\} = \int_J \{m_\lambda^{-i}(t_i) - y_i\}^2 dt, \tag{4.15}$$

to see how well $m_\lambda^{-i}(t_i)$ predicts $m_\lambda^{-i}(t_i)$. The cross-validation score $CV(\lambda)$ is computed by summing up the square errors over all n observations.

$$CV(\lambda) = \sum_{i=1}^{n} \{m_\lambda^{-i}(t_i) - y_i\}^2 dt. \tag{4.16}$$

The smaller the value of $CV(\lambda)$, the better the performance of λ as measured by cross-validation score. Hence, minimizing 4.16 with respect to λ gives the optimal smoothing parameter.

Since cross-validation is very time consuming, and leads to under-smoothing the data so often, the generalized cross-validation criterion has been introduced by Eubank (1988). Here, the value of λ is chosen to minimize

$$GCV(\lambda) = \frac{nSSE(\lambda)}{(n - df_\lambda)^2} = \frac{n}{n - df_\lambda}\hat{\sigma}^2(\lambda) \tag{4.17}$$

where $SSE(\lambda) = \{Y - \Phi(t)\hat{C}\}^\top W\{Y - \Phi(t)\hat{C}\}$ and $\hat{\sigma}^2(\lambda) = SSE(\lambda)/(n - df_\lambda)$. W is the weight matrix. df_λ is the effective number of parameters of degrees of freedom that $\hat{X}(t) = \Phi(t)\hat{C}$ uses in estimating $X(t)$. Heckman and Ramsay (2000) define:

$$df_\lambda = tr(S_{\Phi,\lambda}) + \text{ number of estimated parameters} \tag{4.18}$$

where $S_{\Phi,\lambda}$ is the smoothing matrix as defined in the last section. Usually, the numerator of GCV is small (i.e.,$\Phi(t)\hat{C}$ is close to interpolating the data) when the denominator is small (when df_λ is close to n). Thus minimizing GCV means fitting the data well with few parameters, see Heckman and Ramsay (2000).

The XploRe quantlet `bfacv` and `bfagcv` will be introduced in the section 4.6. These quantlets are used to calculate the CV or GCV of different smoothing parameter λ.

4.6 The Quantlet Bfacv and Bfagcv

`cvl,L= bfagcv(y,argval,fdbasis{,Lfd{,W{,lambda}}})`
Calculate the generalized cross-validation score for smoothing parameter lambda

This quantlet uses 4.16 to compute the GCV score. The syntax of the quantlet is:

`cvl,L= bfagcv(y,argval,fdbasis{,Lfd{,W{,lambda}}})`

with input parameters:

`y` - a vector of (observed) functional values used to compute the functional data object.

`argvals` - vector matrix of argument values.

`basisfd` - an fdbasis object

`Lfd` - Identifies the penalty term. This can be a scalar (≥ 1) containing the order of derivative to penalize, or an LDO object if an LDO with variable coefficients. When applying an LDO with constant coefficients, Lfd is a $(r \times 2)$ matrix, where the first column contains the coefficients, the second one the orders of derivatives. The default value is Lfd = 2.

`W` - Weight matrix. The default value is the identity matrix.

`lambda` - Parameter for roughness penalty smoothing. If lambda is not specified it will be estimated by the data.

When lambda is not given, it will be calculated by

$$\lambda = 10^{-4}\frac{tr\{\phi(t)^\top \Phi(t)\}}{tr\mathtt{R}} \tag{4.19}$$

Lambda will also be returned as the output parameter L.

```
gcv,L= bfagcv(y,argval,fdbasis{,Lfd{,W{,lambda}}})
```
Calculate the generalized cross-validation score for smoothing parameter lambda

This quantlet uses 4.17 to compute the GCV score. The syntax of the quantlet is:

`gcv,L= bfagcv(y,argval,fdbasis{,Lfd{,W{,lambda}}})`

with input parameters:

`y` - $p1 \times p2 \times p3$ array of (observed) functional values used to compute the functional data object. $p1$ is the number of observed values per replication, $p2$ is the number of replications, $p3$ is the number of variables.

`argvals` - $(n \times m)$ matrix of argument values, where $m = 1$ or $m = p2$, respectively. If $m = 1$ the same arguments are applied to all replications and variables. Otherwise it must hold that $m = p2$, then each replication is applied to a certain vector of arguments.

`basisfd` - an fdbasis object

`Lfd` - Identifies the penalty term. This can be a scalar (≥ 1) containing the order of derivative to penalize, or an LDO object if an LDO with variable coefficients. When applying an LDO with constant coefficients, Lfd is a $(r \times 2)$ matrix, where the first column contains the coefficients, the second one the orders of derivatives. The default value is Lfd = 2.

`W` - Weight matrix. The default value is the identity matrix.

`lambda` - Parameter for roughness penalty smoothing. If lambda is not specified it will be estimated by the data for each replication separately.

As this quantlet will use `data2fd` (see Ulbricht (2004))to estimate the coefficients, the syntax of it is in consistency with `data2fd`. The difference is that it does not return an fd object, but the GCV score with respect to smoothing parameter lambda. lambda will also be returned as the output parameter L. When lambda is not given, it will be calculated by 4.19 by using the input parameters y and argvals. If necessary, lambda is computed individually for each replication. If lambda should be zero, then this must be given as input parameter. Note, if the number of basis functions is greater than the number of observations of a single replication, lambda will be estimated by the above formula in any case, even if the corresponding input parameter is set equal to zero. Otherwise (4.17) could not be used because the outer product of the basis matrices in singular in this case. And the value of the estimated lambda will be output, too.

The input parameter y can be a vector, matrix or three-dimensional array, respectively. The missing value problem is processed with the same method as in `data2fd`. When some observations at some arguments are missing just insert `NaNs`. This is also the case when the number of observations vary between the replications. The algorithm filters out the `NaNs`. The input parameter `argvals` cannot contain `NaNs`, otherwise the `getbasismatrix` quantlet need to be readjusted in order to handle `NaNs`. If a functional value is missing for a certain argument, then just give an arbitrary value to the argument. The algorithm will filter out the pair of values in any case. It takes more computation time when `NaNs` are included in the functional data input because the compiler need to calculate the coefficients for each replication separately, see Ulbricht (2004).

We will use some minimization method such as Brent's method or Golden section search, to choose optimal λ.

5 The Implementation of Penalized B-splines on Financial Market Data

5.1 Construction of Model

In this chapter, we will compare the estimation results for different value of smoothing parameters, and choose smoothing parameter λ by cross validation or generalized cross validation. The algorithm of smoothing IV-Strike function includes following steps:

- Define a sequence of non-decreasing knots. When dealing with strike, we use distinct strike prices as the break points. When using moneyness as argument, we set non-decreasing knots every 5 points of sorted moneyness.
- generate basis matrix Φ on the vector of knots;
- Define the penalty operator;
- Choosing the smoothing parameter;

As we have obtained the smoothing function, now we will use it to get SPD. The algorithm is presented as follow:

- Generate a grid on the range of strike price.
- Using the grid points generated from last step as argument values and insert it into the IV smile function. calculate the estimated volatilities and first derivatives as well as second derivatives.
- Using median of stock price as S_t, insert the estimated volatilities, first derivatives, second derivatives into 2.18, and calculate SPD.

5.2 Volatility Smile as Function of Strike

In this section, we will use B-spline smoothing method with constant coefficients LDO to smooth the implied volatility function with respect to strike level. With this function we can get the estimated IV, its first and second derivatives with respect to strike. Insert these data into the SPD calculation function 2.18, using the quantlet `spdcal`, we can finally get the SPD curve.

The data set in table 2.1 with the maturity of $\tau = 0.14167$ (51 days) will be used. Our sample contains 368 observations, as the strike price is set by every 50 euro, observations are concentrating on several distinct strike values ranged from 1900 to 3400. Firstly, we'll place a knot on each strike price. The sequence of non-decreasing knots is (1900, 2000, 2300, 2400, 2450, 2500, 2550, 2600, 2700, 2750, 2800, 2850, 2900, 2950, 3000, 3050, 3100, 3150, 3200, 3300, 3400). We will compare the effect of choosing different order of B-spline.

In figure 5.1, we compare the 4th order and 6th order B-spline. We use $L = D^2$ as penalty operator, smoothing paramter is set to its default value calculated by 4.19. We select 10 observations with different strike prices. The range of strike is from 1900 to 2750.

In figure 5.1, we can observe that the SPD estimated by 6th order B-spline looks smoother, this is reasonable in the sense that the cubic spline function with first and second derivatives at each breakpoints, is claimed to be the lowest-order splines for which the discontinuity in each breakpoints is not visible to the human eye. When we need the 2nd derivatives to calculate SPD, we demand higher order B-splines to permit continuity in breakpoints for higher order derivatives . Because 6-th order B-splines demand relatively more observations, we will use 6th order B-splines to get better look of SPD curve when we have enough observations. When number of observations are small, we will use 4-th order B-splines. Since we don't impose enough penalty on roghness, the estimation is close to just interpolating the data. We can observe sharp drops in all SPD plots, because the IV-strike curve is not smooth enough, which will lead to large difference in estimated first and second derivatives. In next step, by varying the smoothing parameter λ, we will put different weights on curve smoothing, and different IV-strike curves and SPD curves will be plotted accordingly.

Figure 5.2, 5.2, 5.2 use the classical roughness penalty, $L = D^2$. When $\lambda = 10^7$, we can't obtain the enough smoothness, when $\lambda = 10^9$, volatility smile is over smooth and biased, this behavior is explainable in that the trade off between bias and variance.

The penalty on the second derivative of the function tends to force the curve to a straight line, but the iv smile is not a straight line. Based on this argument, it is suggested to use $L = D^3$ as a substitution. Next we do the experiment with the 3rd order LDO, $L = D^3$ on the same data set. We have to use extremely large value of λ

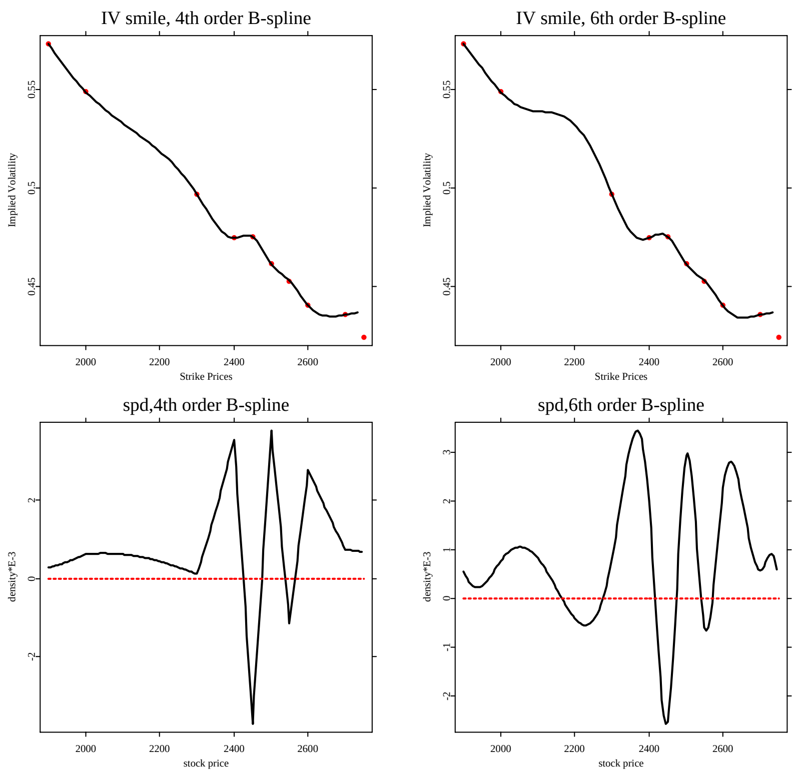


Figure 5.1: IV-Strike, DAX call option on 02-25-03, tau=0.14167, Top: IV smile. Bottom: SPD, red horizon line represents 0. Left: 4th order b-spline, right: 6th order B-spline

`test46.xpl`

to smooth the curve. The larger the λ is, the smoother the SPD is.

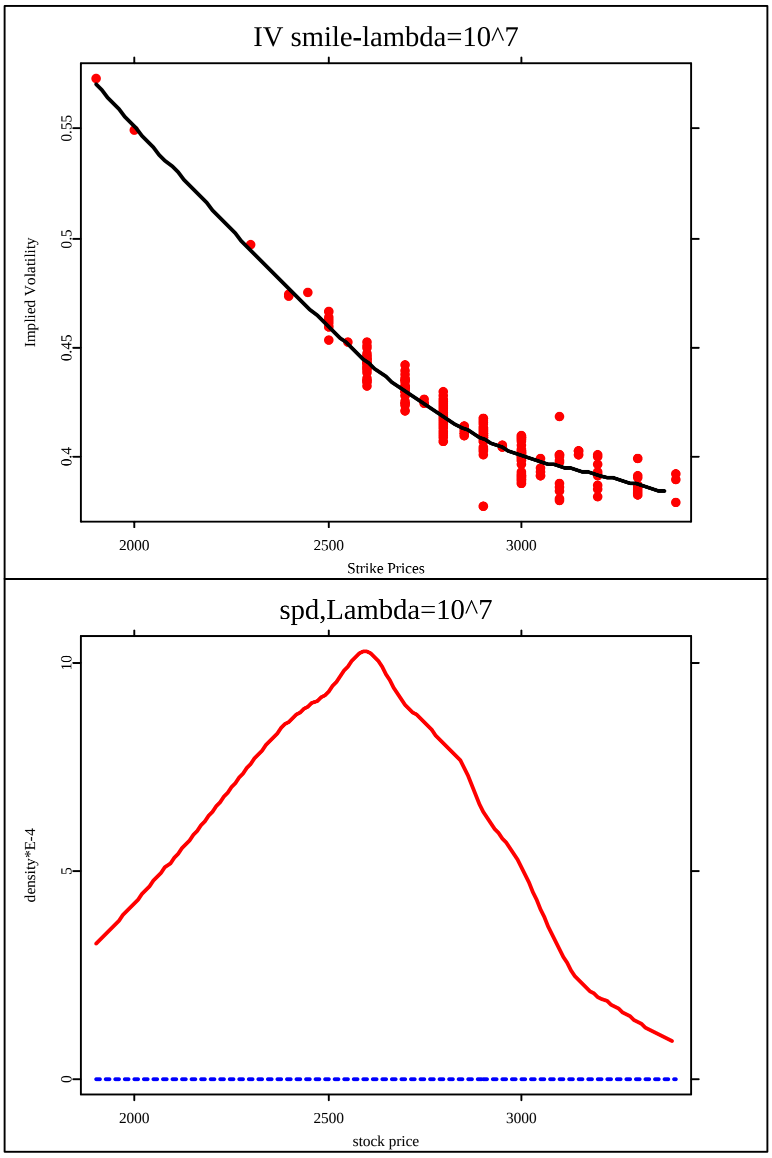


Figure 5.2: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^7$, The blue dashed horizontal line represents 0. spd2.xpl

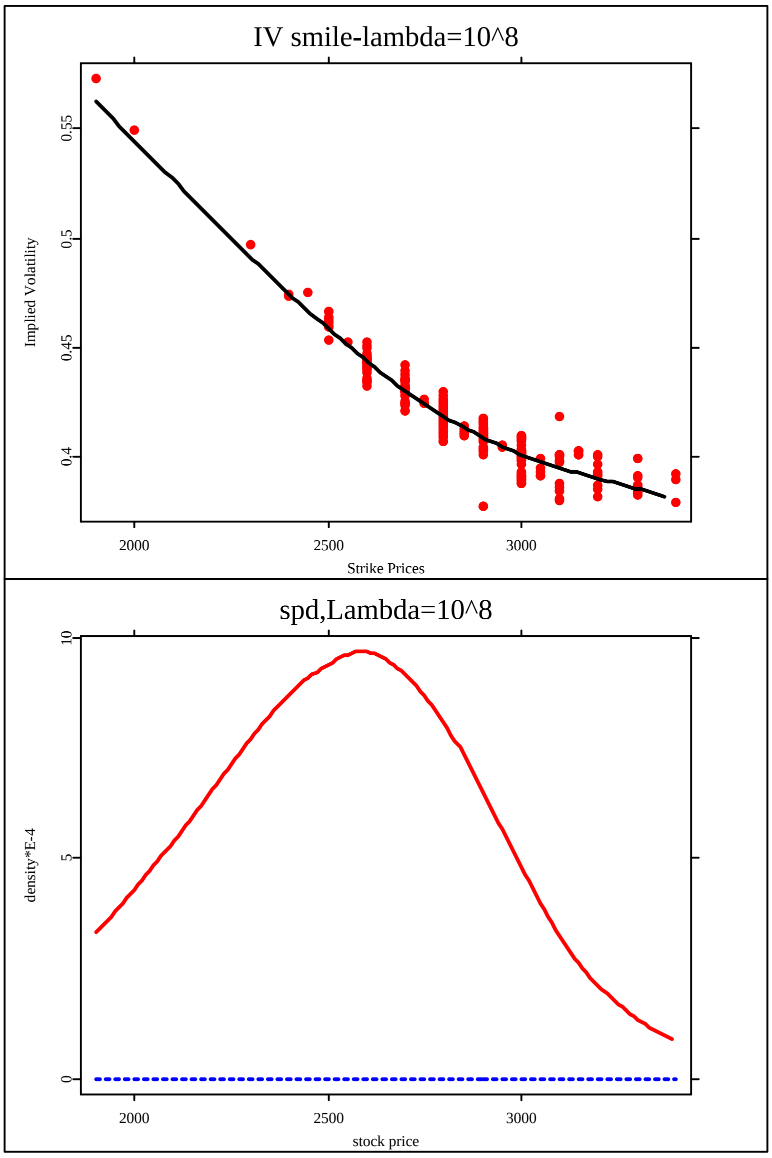


Figure 5.3: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^8$, The blue dashed horizontal line represents 0.

spd2.xpl

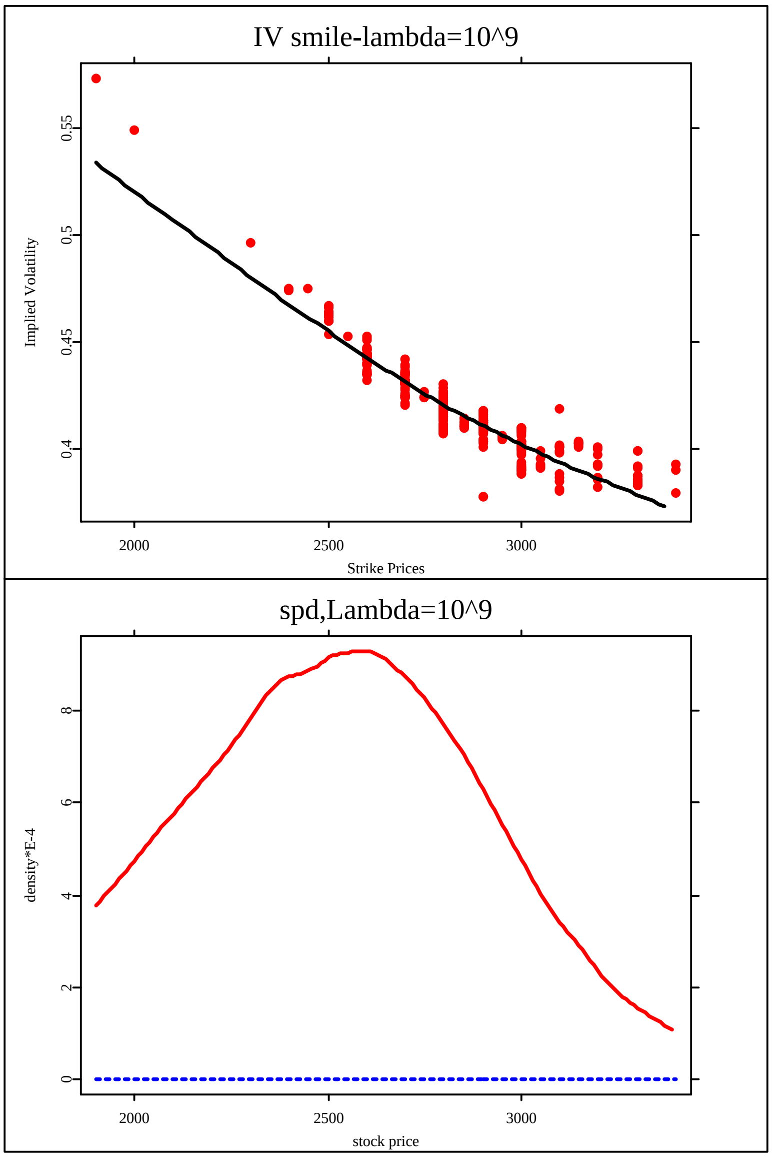


Figure 5.4: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^9$, The blue dashed horizontal line represents 0.

spd2.xpl

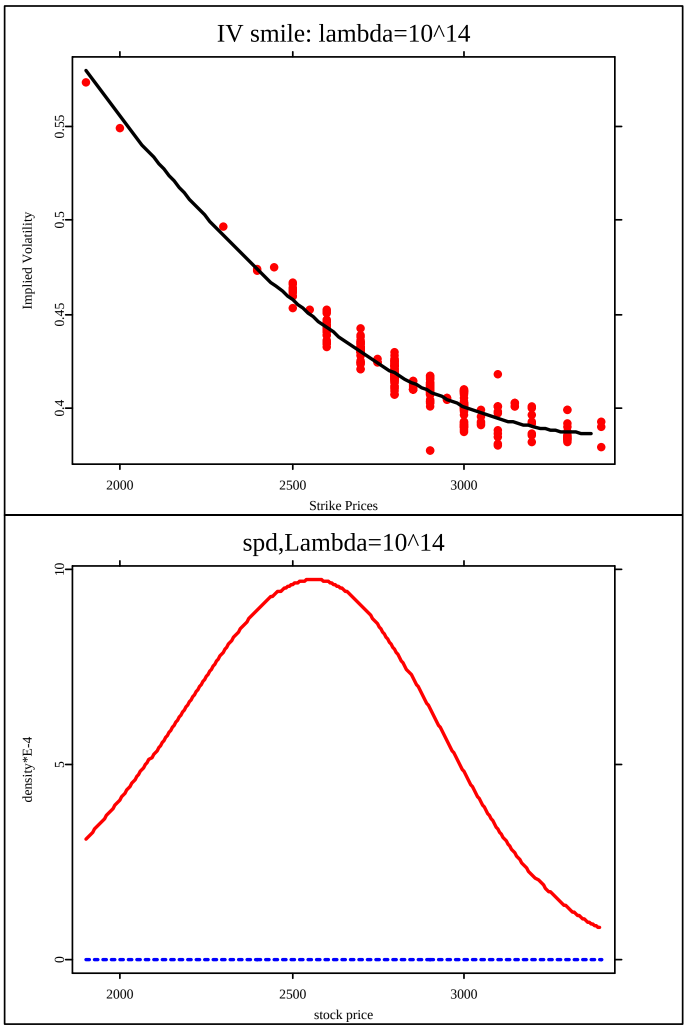


Figure 5.5: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^14$, The blue dashed horizontal line represents 0.

spd3.xpl

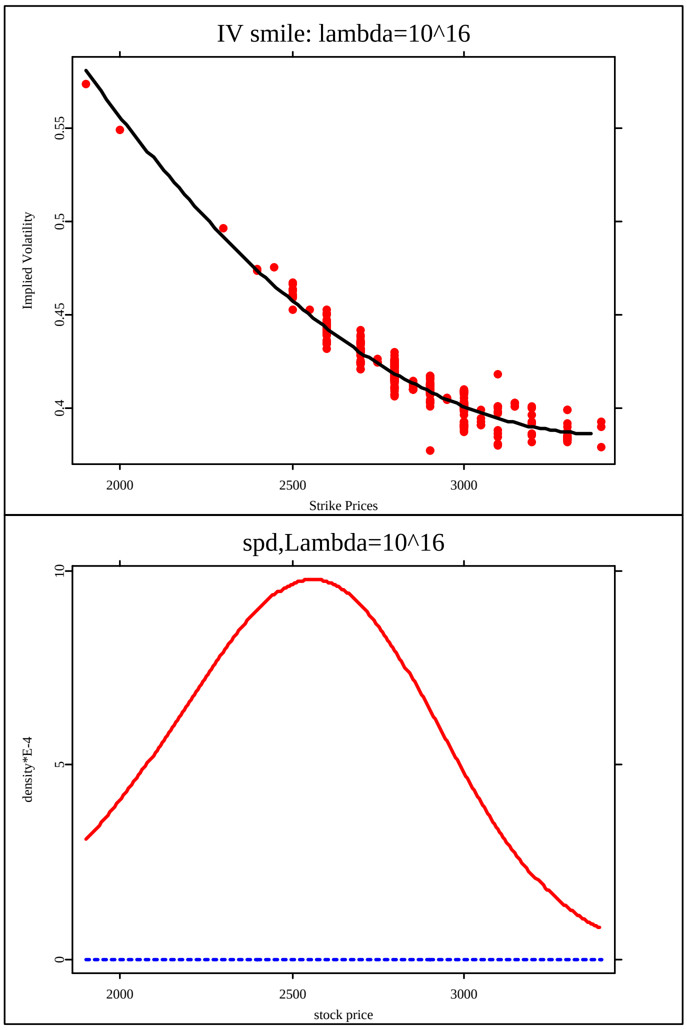


Figure 5.6: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 10^16$, The blue dashed horizontal line represents 0.

spd3.xpl

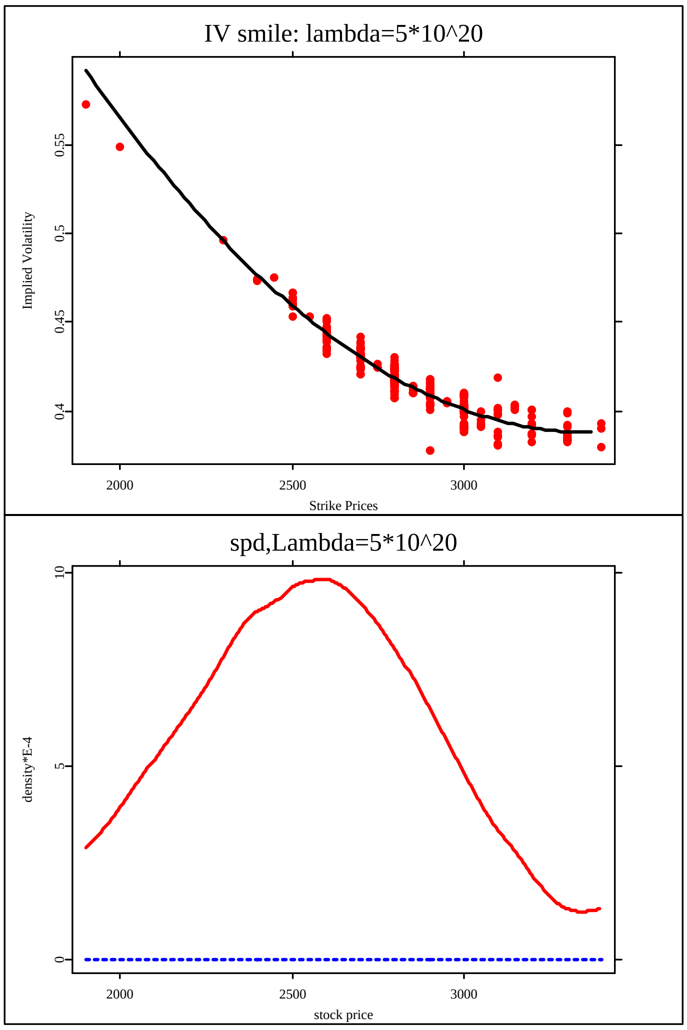


Figure 5.7: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$, $\lambda = 5 \times 10^20$, The blue dashed horizontal line represents 0.

spd3.xpl

After the subjective method on choosing λ, we will choose λ based on the data set, as the method introduced in 4.5. The cross-validation will be very time consuming since our sample contains 368 observations. We will use GCV to make this choice. GCV score of different smoothing parameter λ is computed, and the selection will be made based on this criterion again. Table 5.1 provides the GCV score according to each λ with respect to different linear differential operator.

$L = D^2$		$L = D^3$	
λ	GCV	λ	GCV
10^7	0.0015045	10^{14}	0.0014791
10^8	0.0014863	10^{16}	0.001478
10^9	0.0014931	5×10^{19}	0.001478

Table 5.1: GCV score comparison testgcv23.xpl

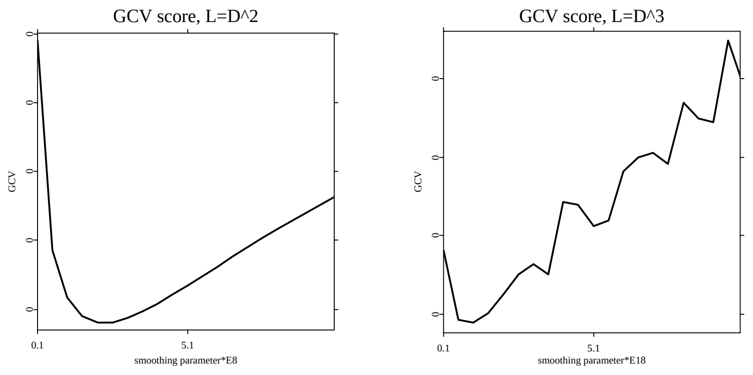


Figure 5.8: GCV score with respect to smoothing parameter λ, left panel: $L = D^2$, right panel: $L = D^3$. testgcv.xpl

From table 5.1 we can also speculate that for $L = D^2$, the λ that minimize GCV function 4.17 can be located in $[10^7, 10^9]$. Using $(10^7, 10^8, 10^9)$ as a bracket, we can find this best value by using brent's method or golden section search. XploRe quantlet `nmbrent` or `nmgolden`can be applied for minimization.

Below is the result returned by using `nmbrent`:

Contents of lmin.xmin[1,] 2.3216e+08

Contents of lmin.fmin.gcv[1,] 0.001484

`minil.xpl`

By set $\lambda = 2.3216 \times 10^8$ we get figure 5.9. The upper left figure is IV-Strike curve, and the upper right figure is the estimated SPD, it's well above the 0 level. We also display the log-normal distribution simultaneously for comparison. The mean and variance of log-normal distribution is equal to function 2.7, with volatility and stock price calculated by mean. The bottom left figure is the first derivative of volatility respect to strike, and the bottom right figure displays the first derivative together with its no arbitrage bounds, refer to 2.6. We also calculate the integral of the SPD. Since we don't have an explicit function for the estimated SPD curve, the integral can't be carried out directly. We can only do it in an approximate manner. The substitutional method is: divide the strike price line into several small equal intervals, denote h as the width of each interval, then we have $n = \frac{max(K)-min(K)}{h}$ intervals. Denote x_i as the start point of each interval, and $f(x_i)$ is the estimated density value of x_i. The approximate integral can be calculated analytically as:

$$\int_{min(K)}^{max(K)} f(x)dx \approx \sum_{i=1}^{n} x_i f(x_i) \tag{5.1}$$

The more the number of interval is, the more precisely is the approximation result. for SPD curve of 5.9. We have a stock price range of $[1900, 3400]$, divide it into 1500 intervals with each bin width as 1, and calculate the approximated integral with quantlet `spdgcv2`. The result is 0.89325 .

The IV smile we get from 5.9 is biased. This is because the usage of $D = L^2$ tends to force the curve to a straight line. We also choose the best smoothing parameter when $L = D^3$ by minimizing the GCV score. $\lambda = 9.1356 \times 10^{17}$, GCV score= 0.001478 , the integral is 0.8958. We also plot the second derivative of IV with respect to strike, as in the bottom left panel. In Bottom right panel, we plot first derivative together with its no arbitrage bounds. As figure 5.10 shows, when using $L = D^3$, the IV smile matches the data better than using $L = D^2$, especially in the initial part where there is sparse observation. The first derivative of IV lies just within the arbitrage free bounds.

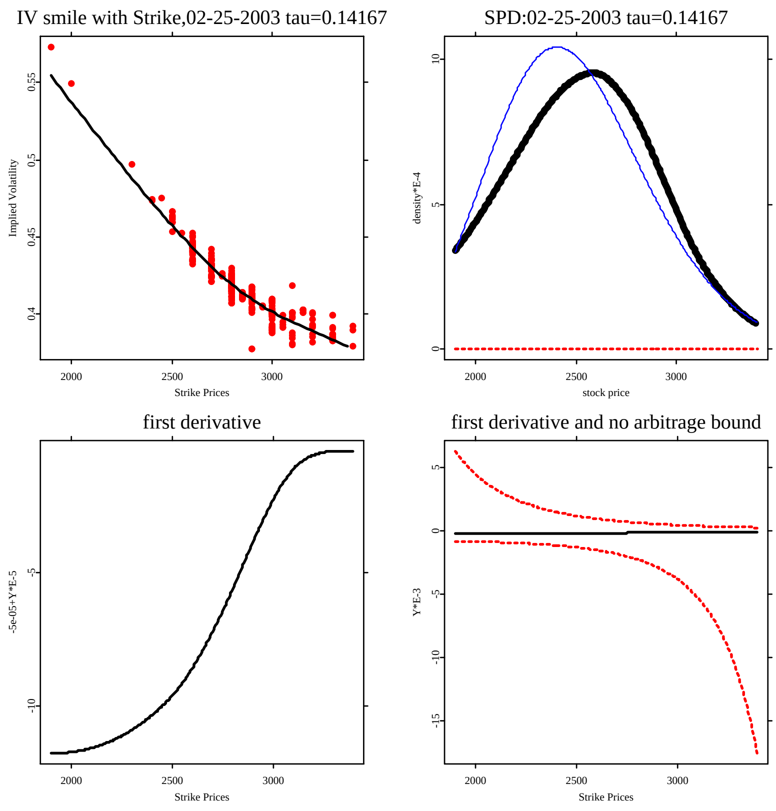


Figure 5.9: DAX call option, 02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with strike. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative of volatility with respect to strike. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line). spdgcv2.xpl

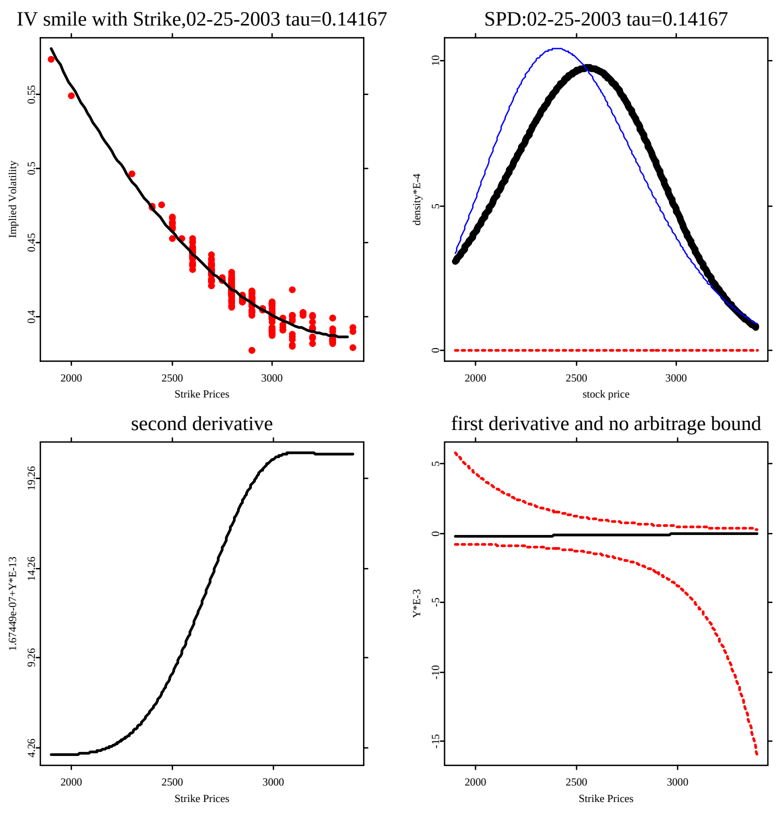


Figure 5.10: DAX call option, 02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with strike. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). spdgcv3.xpl

5.3 Multiple Observations

As we have mentioned previously, due to the 50 euro interval of strike, our data set are consist of multiple observations of volatilities at a finite vector of strike prices, range from 1900 to 3400. This problem can be solved by using moneyness as a measure, or modify our set-up to incorporate this characteristic.

One possible solution for multiple observations is using just ”closing prices” for each strike. That is to mean, for each strike, we select the option that traded at the latest moment of the day. It is convincible since the closing price contains more information and is thus more reliable. This procedure will largely decrease the sample size and may cause the estimation less accurate and complete.

We will use the same data set as in table 2.1, the DAX call option with the maturity of $\tau = 0.14167$. The 9th column of the data set contains the trade time, it is the seconds from midnight. After modification, we have a sample contains 21 observations, each strike will be used as a distinct knot. Since we have very few observations now, we will use 4th order B-splines, and the smoothing parameter will be chosen by cross-validation this time. Figure 5.11 plots CV score with respect to different smoothing parameters.

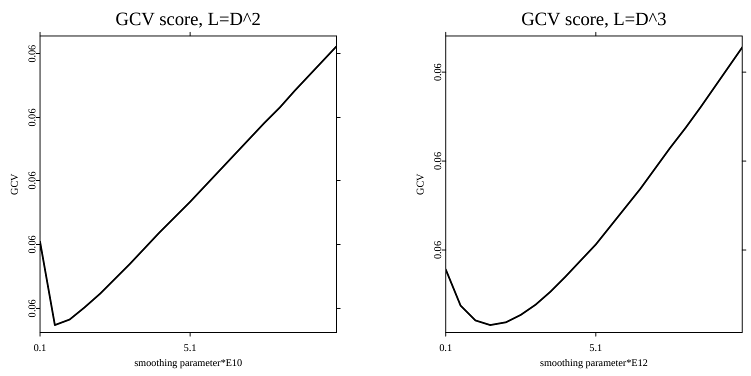


Figure 5.11: CV score with respect to smoothing parameter λ, left panel: $L = D^2$, right panel: $L = D^3$.

testcv.xpl

Figure 5.12 and 5.13 are plotted with penalty operator $L = D^2$ and $L = D^3$ respectively. Smoothing parameter is chosen by CV. λ chosen by cross-validation oversmooth the IV function when $L = D^2$ is employed. The IV smile fits the observation quite better by employing $L = D^3$.

	$L = D^2$	$L = D^3$
λ	6.4279×10^9	1.6521×10^{12}
CV	0.064276	0.064635
Integral	0.89382	0.89335

Table 5.2: Parameters, Closing Day Data

`minilcd.xpl`

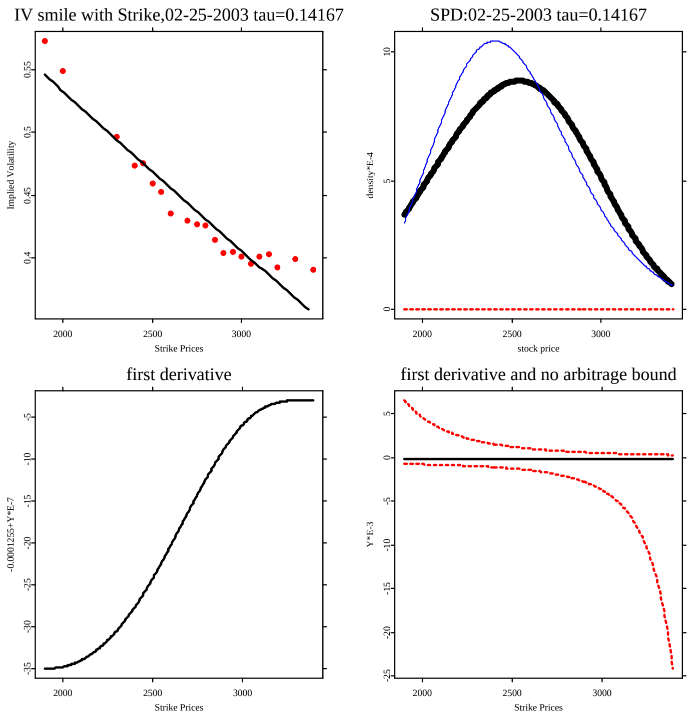


Figure 5.12: DAX call option, $L = D^2$. Upper left panel: IV smile. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative of volatility. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line).

`spdcd2.xpl`

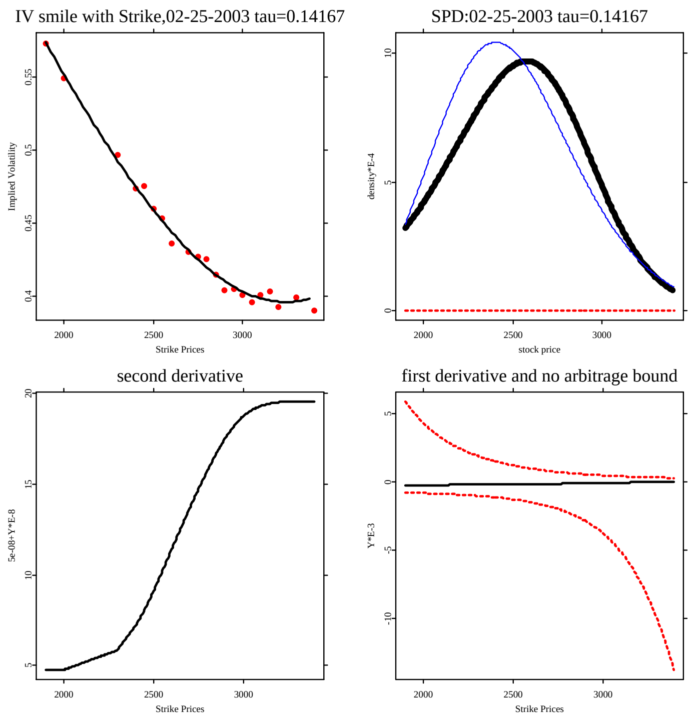


Figure 5.13: DAX call option, $L = D^3$. Upper left panel: IV smile. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivatives. Bottom right panel: the first derivative (black line) and its no arbitrage bounds (red dashed line). `spdcd3.xpl`

5.4 Volatility Smile as Function of Moneyness I

In this section, we will use IV smile with moneyness to estimate SPD. The usage of moneyness can avoid the problem of multiple observations. We use 6-th order B-spline and place a knot for every 10 grids fo moneyness. From figure 5.14, 5.15, we can find another advantage of using moneyness as argument is that we need not impose huge weight on roughness penalty to balance the bias and variance. The smoothing parameter λ is chosen by GCV. There is no big difference with two penalty operators, $L = D^2$ and $L = D^3$ in shaping SPD, and both of them work well in smoothing the IV function when there are dense observations. When $L = D^2$, the IV smile curve is biased at the initial part where there are sparse observations. $L = D^3$ has a superiority in matching data in the area with sparse observations. Table 5.3 offers smoothing parameters chosen by GCV and GCV scores.

	$L = D^2$	$L = D^3$
λ	0.0074169	0.1155
GCV	0.00061656	0.00061233

Table 5.3: Parameters

minilm.xpl

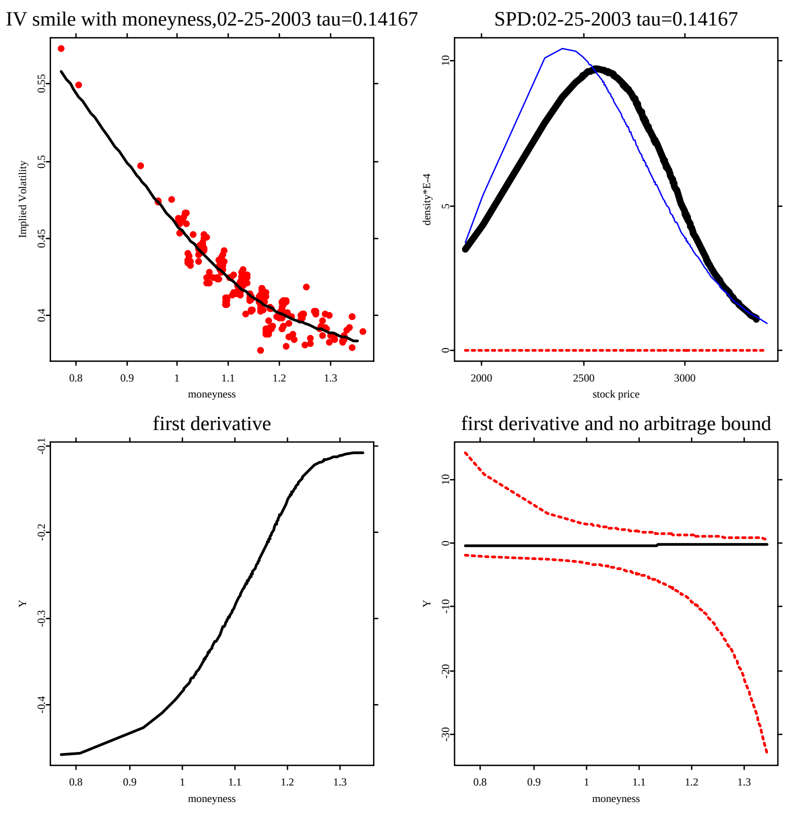


Figure 5.14: DAX call option,02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). spdm2.xpl

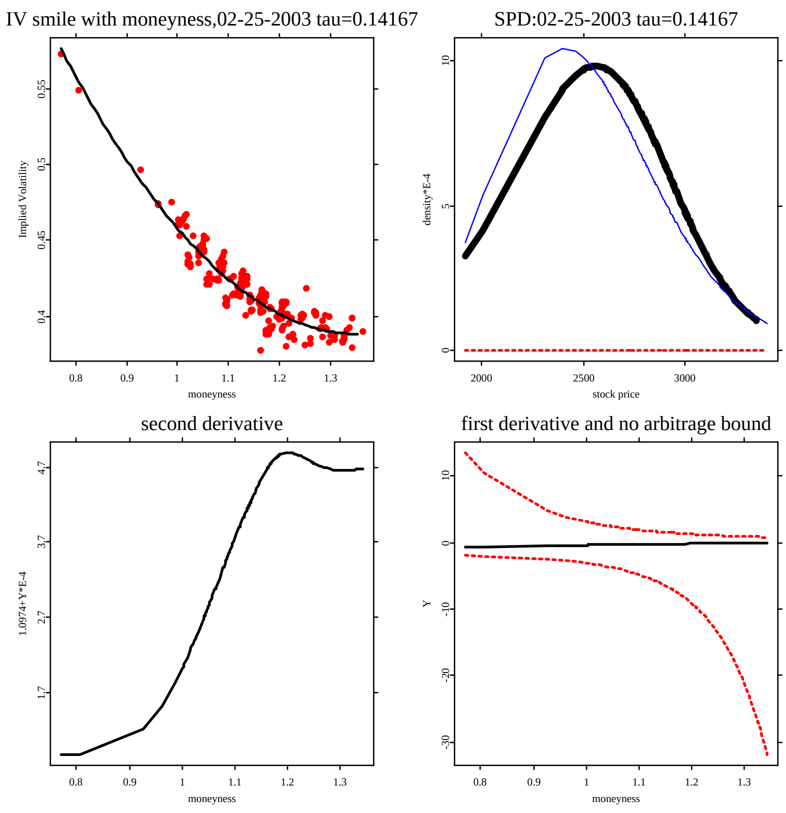


Figure 5.15: DAX call option,02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). spdm3.xpl

5.5 Volatility Smile as Function of Moneyness II

In all last section, we make the estimation with original data from financial market. Multiple observations is one problem when using strike as measure. Moneyness could be a good substitution, but observations also inevitablely concentrate on neighborhood area of several values. See IV smile plot in figure 5.14 and 5.15, the right tail part is crowded by dense observations. And we also want to take strike and transaction time into consideration even using moneyness as a measure.

Denote the i-th observations of the strike price by K_i, the corresponding volatility $\sigma_i = \sigma(K_i, t)$ and corresponding moneyness $\kappa_i = \kappa(K_i, t)$, here t denotes transaction time, its the seconds from midnight. In practice, volatilities and moneynesses are repeated for a small number of distinct strike prices. $\sigma = (\sigma_1, ..., \sigma_n)^\top$ and $\kappa = (\kappa_1, ..., \kappa_n)^\top$ are the vector of volatility and moneyness respectively. The corresponding vector of the strike prices has the following structure:

$$K = \begin{pmatrix} K_1 \\ K_2 \\ \vdots \\ K_n \end{pmatrix} = \begin{pmatrix} k_1 1_{n1} \\ k_2 1_{n2} \\ \vdots \\ k_n 1_{np} \end{pmatrix} \tag{5.2}$$

where $k_1 < k_2 < ... < k_p$, $n_j = \sum_{i=1}^n I(K_i = k_j)$ with $I(.)$ denoting the indicator function and 1_n a vector of ones of length n.

Now we will construct the model by following way:

- Divide the sample data into p groups by different strikes
- Set every 30 minutes (1800 seconds) as a interval, in each group, divide the data again by this time interval.
- Now calculate the median of moneyness and volatility for each sub-group
- Use medians of moneyness and volatility as a new data set and apply smoothing techniques to this new data set

Run this procedure on XploRe. We get fugure 5.16 and 5.17. We use 6-th order B-spline and place a knot for every 5 grids fo moneyness. smoothing parameter is chosen by GCV. Table 5.3 offers smoothing parameters chosen by GCV and GCV scores.

	$L = D^2$	$L = D^3$
λ	0.0014915	2.8473
GCV	0.00010385	0.00010362

Table 5.4: Parameters

minilmo.xpl

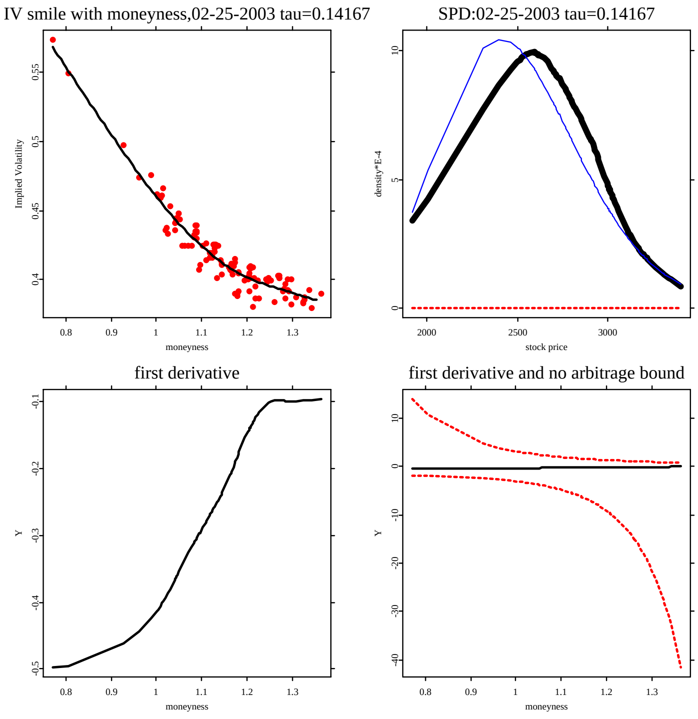


Figure 5.16: DAX call option,02-25-03, $\tau = 0.14167, L = D^2$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: first derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line).

spdmmo2.xpl

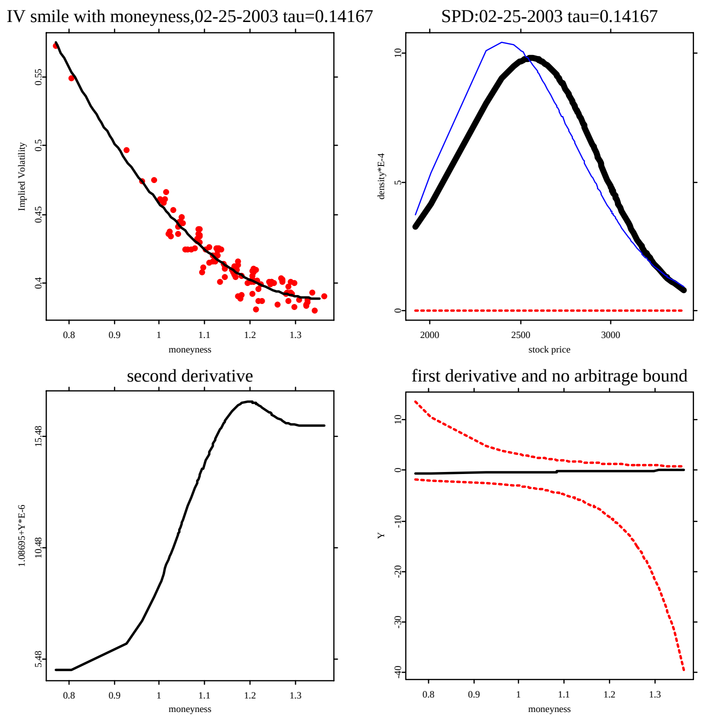


Figure 5.17: DAX call option,02-25-03, $\tau = 0.14167, L = D^3$. Upper left panel: IV smile with moneyness. Upper right panel: SPD (black thick line), log-normal distribution (blue thin line), red horizon line represents 0. Bottom left panel: second derivative. Bottom right panel: first derivative (black line) and its no arbitrage bounds (red dashed line). spdmmo3.xpl

5.6 Conclusions

In this chapter, we use penalized B-splines to smooth the IV function. We compared the estimation using two parameterizations: the penalty operator and the smoothing parameters. Our estimations are carried out through two lines: strike and moneyness. First, when strike is used as argument, the major problems are repeated observations on discrete strike value. To avoid this repeated observations problem, we estimated the model using closing prices. This approach is also flawed due to the small sample size. The application of moneyness as IV function argument can solve the problem of multiple observations. Meanwhile, we need not to impose huge weight on roughness penalty to force the curve smooth enough. The last estimation method combines strike and moneyness together with transaction time. The data is grouped by strike and transaction time, and the group median forms a new data set, finally we use this reconstructed model to smooth IV smile and calculate SPD.

When $L = D^2$ is used as roughness penalty, the IV smile function is biased in the wings. This is due to the unequal distribution of IV. This becomes more obvious for the extremely large smoothing parameter. When $L = D^3$ is used as penalty operator, this effect is less present, even for larger smoothing parameters the bias remains small.

We use cross-validation or GCV to choose smoothing parameter. When the second derivative of function is penalized, the smoothing parameters chosen by CV or GCV tends to yield visible bias in the wings of smile function, but they can yield good visual effect of IV smile when $L = D^3$. The heavy penalty on third derivatives will force the first derivative of function to a straight line, which makes IV smile resemble a quadratic parabola. This effect deviates from the empirical observation. The over smoothing effect remains.

All of the discussed models are successful to produce non negative and smooth SPD. The first derivatives of IV smile function are exactly located within their no-arbitrage bounds.

6 Summary and Outlooks

The aim of this paper is to implement different smoothing techniques on IV smile function and compute the state price density. The Nadaraya-Watson and Local polynomial estimator are briefly introduced. Nadaraya-Watson estimator is not recommended here because of its limitation on fitting unequally distributed data and the inability of estimating derivatives directly. Higher order local polynomials avoid those problems and is more adaptive for our goal.

The emphasis of this paper is put on the application of penalized B-splines to IV smile. The imposing of roughness penalty on derivatives differentiates this method from other smoothing techniques. By the choice of penalty operator, the shape of the curve can be controlled; and the smoothing parameter can be adjusted to balance the smoothness of the curve and the fitness of the curve to data. These two instruments make this method more flexible and fruitful in fitting IV smile function with some degree of smoothness. We have used $L = D^2$ and $L = D^3$ as roughness penalty operator. $L = D^3$ has better performance in wings of IV smile function where the observations are sparse. Like higher order local polynomials, we can estimate derivatives from the B-splines directly. This advantage is notable in calculating SPD.

The implied volatility smile should be under certain constraints, as introduced in 2.6. The penalized B-splines can be extended to fulfill those constraints. The application of special designed LDO may yield other interesting results in IV smoothing and the calculation of SPD, and it might be another aspect for further extension.

7 Appendix

7.1 XploRe quantlet lists

7.1.1 Statistical Tools

Bfacv

```
proc (cvl,L) =bfacv(y, argvals, basisfd, Lfd,W,lambda)
; -----------------------------------------------------------
; Library     fda
; -----------------------------------------------------------
; See_also    data2fd, createfdbasis, evalfd, getbasismatrix, bfagcv
; -----------------------------------------------------------
; Macro       bfacv
; -----------------------------------------------------------
; Description: calculate CV score
; -----------------------------------------------------------
library("fda")
  if (exist (Lfd) == 0)
    Lfd = 2
  endif

  if (exist (lambda) == 0)
    lambda = 0
    boollambda = 0
  else boollambda = 1
  endif

    if (exist (W) == 0)
    W = diag (matrix (rows (y)))
  endif

  if (dim (#(W)) == 1)
    W = W .* diag (matrix (rows (y)))
  endif


error (sum (!#(argvals <= Inf)) != 0, "ARGVALS cannot contain NaNs or
  INFs")
```

```
error (rows (dim (lambda)) != 1 || rows (lambda) != 1, "LAMBDA must be
  a scalar")
error (rows (dim (y)) != 1, "y must be a vector")
error (lambda<0, "LAMBDA must be non negative")

 if (boollambda == 1 && lambda == 0)
    inprodmat = unit (basisfd.nbasis)
  else
    inprodmat = inprod (basisfd, basisfd, Lfd, Lfd)
    if (basisfd.nbasis >= rows (y))
; add a very small multiple of the identity to inprodmat:
      inprodmat = inprodmat + 1e-10 * max (#(inprodmat)) * unit (rows (
        inprodmat))
    endif
  endif

  xx=rows(y)
  basismat = getbasismatrix (argvals, basisfd, 0)
se=matrix(xx)
 cvl=matrix(xx)
 mi=matrix(xx)
  n=rows(argvals)
     i = 1
      while (i <= xx)
; begin: boollambda
    if (boollambda == 0 || (boollambda == 1 && lambda == 0 && rows (y)
      < basisfd.nbasis))
      lambda = 0.0001 * spur (basismat' * basismat) / spur (inprodmat)
fdo=data2fd(y, argvals, basisfd, Lfd, W)   ;lambda is set to equal to
  the default value of lambda in data2fd
   else
fdo=data2fd(y, argvals, basisfd, Lfd, W,lambda)
    endif

eva=evalfd(argvals,fdo,0)
mi[i]=(cumsum(eva)[n-1]-eva[i])/(n-1)
se[i]=(mi[i]-y[i])^2
cvl=cumsum(se)[n-1]
     i=i+1
      endo
 L=lambda
  endp
```

Bfagcv

```
proc (gcv,L) =bfagcv(y, argvals, basisfd, Lfd, W,lambda)
; ---------------------------------------------------------
; Library     fda
; ---------------------------------------------------------
; See_also   data2fd, createfdbasis, evalfd, getbasismatrix, bfacv
; ---------------------------------------------------------
; Macro      bfagcv
; ----------------------------------------------------------
; Description calculate GCV score
; ----------------------------------------------------------
library("fda")
  if (rows (dim (y)) == 3)
    xx = dim (y)[3]
  else xx = 1
  endif

  if (exist (Lfd) == 0)
    Lfd = 2
  endif

  if (exist (lambda) == 0)
    lambda = 0
    boollambda = 0
  else boollambda = 1
  endif

  if (exist (W) == 0)
    W = diag (matrix (rows (y)))
  endif

  if (dim (#(W)) == 1)
    W = W .* diag (matrix (rows (y)))
  endif

  error (dim (W) != dim(y)[1], "W must be quadratic with dimension
    equal to the rows of Y")
  error (sum (!#(argvals <= Inf)) != 0, "ARGVALS cannot contain NaNs or
     INFs")
 error (rows (dim (lambda)) != 1 || rows (lambda) != 1, "LAMBDA must be
    a scalar")
   error (lambda<0, "LAMBDA must be non negative")

 if (boollambda == 1 && lambda == 0)
    inprodmat = unit (basisfd.nbasis)
  else
    inprodmat = inprod (basisfd, basisfd, Lfd, Lfd)
    if (basisfd.nbasis >= rows (y))
; add a very small multiple of the identity to inprodmat:
      inprodmat = inprodmat + 1e-10 * max (#(inprodmat)) * unit (rows (
        inprodmat))
    endif
  endif


  basismat = getbasismatrix (argvals, basisfd, 0)
  SSE=matrix(cols (y), xx)
  df=matrix(cols (y), xx)
  gcv=matrix(cols (y), xx)
  n=rows(argvals)


 if (sum (!#(y <= Inf)) == 0)  ; case: functional data is a vector or matrix, no missing values!!!


      i = 1
      while(i<=cols(y))
      j = 1
      while (j <= xx)
; begin: boollambda
    if (boollambda == 0 || (boollambda == 1 && lambda == 0 && rows (y) < basisfd.nbasis))
      lambda = 0.0001 * spur (basismat[,,,i]' * basismat[,,,i]) / spur (inprodmat)
fdo=data2fd(y, argvals, basisfd, Lfd, W)   ;lambda is set to equal to the default value of lambda in data2fd
   else
fdo=data2fd(y, argvals, basisfd, Lfd, W,lambda)
    endif

coef = fdo.coef
  n1= fdo.basisfd.nbasis

 SSE[i,j] = (y[,i,j]-basismat[,,,i] * coef[,i,j])' * W *(y[,i,j]-basismat[,,,i] * coef[,i,j])
   smoothmat = basismat[,,,i]* inv (basismat[,,,i]' * W * basismat[,,,i] + lambda * inprodmat) * basismat[,,,i]' * W
     df[i,j]=spur(smoothmat)+n1
        gcv[i,j]=n*SSE[i,j]/(n-df[i,j])^2
       j = j + 1
      endo
      i=i+1
      endo

 else  ;case: functional data is vector or matrix, including missing values
    j = 1
    while (j <= xx)
  i = 1
         while (i <= cols (y))
          kk = 1 : rows (y)
          kk = paf (kk, y[kk,i,j] <= Inf)
          yy = y[kk,i,j]
  ; reformulate lambda:
          if (boollambda == 0 || (boollambda == 1 && lambda == 0 && rows (yy) < basisfd.nbasis))
            lambda = 0.0001 * spur (basismat[kk,,,i]' * basismat[kk,,,i]) / spur (inprodmat)
fdo=data2fd(y, argvals, basisfd, Lfd, W)  ;lambda is set to equal to the default value of lambda in data2fd
 else
```

```
fdo=data2fd(y, argvals, basisfd, Lfd, W,lambda)
endif

coef = fdo.coef
 n1= fdo.basisfd.nbasis

          SSE[i,j]=(yy-basismat[kk,,,i] * coef[,i,j])'* W[kk,kk]*(yy-
            basismat[kk,,,i] * coef[,i,j])
         smoothmat =  basismat[kk,,,i]* inv (basismat[kk,,,i]' * W[kk,
           kk] * basismat[kk,,,i] + lambda * inprodmat) * basismat[kk
           ,,,i]' * W[kk,kk]
         df[i,j]=spur(smoothmat)+n1
         gcv[i,j]=n*SSE[i,j]/(n-df[i,j])^2
              i = i + 1
        endo
      j = j + 1
    endo
  endif
  L=lambda
  endp
```

spdcal

```
proc(fstar)=spdcal(k, sig, sig1, sig2, s, r, tau)
; -----------------------------------------------------------
; Library     finance
; -----------------------------------------------------------
; See_also    spdcalm
; -----------------------------------------------------------
; Macro       spdcal
; -----------------------------------------------------------
; Description calculate the spd by strike and implied volatility
; -----------------------------------------------------------
 error((sum(sum(sig<0)')>=1), "spdcal: Watch out: Some of your
   volatilities are negative!!")
  rk=rows(k)
  st=sqrt(tau)
  ert=exp(r*tau)
  rt=r.*tau

 d1=(log(s/k)+tau.*(r.+0.5.*(sig.^2)))./(sig.*st)
  d2=d1-sig.*st

fstar=(pdfn(d2)./ert).*( (1/(sig.*k.*st))+(2*d1./sig).*sig1)+(pdfn(d2)
  ./ert).*(d1.*d2.*k.*st.*sig1.^2./sig+ k.*st.*sig2)
fstar=fstar.*ert

endp
```

spdcalm

```
proc(fstar)=spdcalm(m, sig, sig1, sig2, s, r, tau)
; ------------------------------------------------------------
; Library     finance
; ------------------------------------------------------------
; See_also    spdcal
; ------------------------------------------------------------
; Macro       spdcalm
; ------------------------------------------------------------
; Description calculate the spd by moneyness and implied volatility
; ------------------------------------------------------------

 error((sum(sum(sig<0)'))>=1), "spdcal: Watch out: Some of your
   volatilities are negative!!")

  st=sqrt(tau)
  ert=exp(r*tau)
  rt=r.*tau
  ft=s.*ert

 d1=(log(s./(m.*ft))+tau.*(r.+0.5.*(sig.^2)))./(sig.*st)
  d2=d1-sig.*st

fstar=pdfn(d2).*( 1/(sig.*(m.*ft).*st)+(2*d1./sig).*sig1./ft)+pdfn(d2)
  .*(d1.*d2.*(m.*ft).*st.*(sig1.^2)./(ft.^2)./sig+ (m.*ft).*st.*sig2./(
  ft.^2))

endp
```

7.1.2 Implemention Examples

I have written a lot of quantlet to compare different results from different parameters, but they are mainly implemented in a similiar way. So I won't list all of them here, but only list some examples to illustrate the procedure and method. The sample data is DAX European opion on 25th, February, 2003, as listed in table 2.1

testgcv

Calculate GCV score of smoothing parameter. Plot the GCV score with respect to different smoothing parameters.

```
library("finance")
library("fda")

proc(gcv)=fdagcvv(y, argvals, bspl, lfd, W, lam)
error (rows (dim (lam)) != 1, "lam must be a vector")
r=rows(lam)
gcv=matrix(r)
i=1
while (i<=rows(lam))
```

```
 gcv[i] =bfagcv(y,argvals, bspl,lfd,W,lam[i])
i=i+1
endo
endp

 x=read("rawdata030225.DAT")        ; read the data
x=x[,1:6]
x=paf(x,x[,6]==1)
x=paf(x,x[,4]>0.14&& x[,4]<0.15)
tau=x[,4]
s=x[,1]
r=x[,3]
k=x[,2]
y=ImplVola(x)       ; calculate ImplVola
smile=k~y
smile=sort(smile)
smile=smile[1:15|17:rows(smile)] ;delete the outlier
tvec=#(1900, 2000, 2300, 2400, 2450, 2500, 2550, 2600, 2700, 2750,
  2800, 2850, 2900, 2950, 3000, 3050, 3100, 3150, 3200, 3300, 3400)
 ; generate a non decreasing sequence of knots, the number of knot is
   decided by the dense of strike parice
 bspl = createfdbasis ("bspline", #(1900, 3400), 25, tvec)
c=grid(10^7, 5*10^7, 21)
cgcv=fdagcvv(smile[,2],smile[,1],bspl,2,1,c)
cgcv=c~cgcv
cgcv=setmask(cgcv,"line")
setsize(400,400)
pic=createdisplay(1,1)
show(pic,1,1,cgcv)
setgopt(pic,1,1,"yvalue",0|1,"xlim",10^7|10^9,"xorigin", 10^7, "title",
  "GCV score, L=D^2", "xlabel","smoothing parameter","ylabel","GCV","
  border",0)

c1=grid(10^17, 5*10^17, 21)
cgcv1=fdagcvv(smile[,2],smile[,1],bspl,3,1,c1)
cgcv1=c1~cgcv1
cgcv1=setmask(cgcv1,"line")
setsize(400,400)
pic1=createdisplay(1,1)
show(pic1,1,1,cgcv1)
setgopt(pic1,1,1,"yvalue",0|1,"xlim",10^17|10^19,"xorigin", 10^17, "
  title","GCV score, L=D^3", "xlabel","smoothing parameter","ylabel","
  GCV","border",0)
```

tempminicd

Find the smoothing parameter by cross-validation.

```
library("finance")
library("fda")
library("nummath")

```

```
proc(ct)=count(x)
;count number of different groups divided by different strike
;and generate a new vector with each element being the group number
error(cols(x)>1, "x must be a vector")
i=1
ct=matrix(rows(x))
while (i< rows(x))
if ( x[i]== x[i+1])
ct[i+1]=ct[i]
else
ct[i+1]=ct[i]+1
endif
i=i+1
endo
endp

proc(subm)=submax(x,n)
;1.divide x by different groups by different values of first colum of x
;2. find out the maximum of n-th colum of x in each group.
;3. select the according observations and constitute a new matrix
ct=max(x[,1]) ;count the number of different values in first column of
  x
subm=matrix(ct,cols(x))       ;matrix of submax
i=1
k=1
while (i<=max(x[,1]) )
xx=paf(x,x[,1]==i)
mm=max(xx[,n])
subm[k]=paf(xx,xx[,n]==mm)
i=i+1
k=k+1
endo
endp

 proc(cvl)=minil(lam)
 x=read("rawdata030225.DAT")        ; read the data
t=x[,9]
x=x[,1:6]
x=x~t
x=paf(x,x[,6]==1)
x=paf(x,x[,4]>0.14&& x[,4]<0.15)
t=x[,7]
tau=x[,4]
s=x[,1]
r=x[,3]
k=x[,2]
y=ImplVola(x[,1:6])
x1=k~t~y
x1=x1[1:7|9:rows(x1)] ;delete outlier
x2=count(x1[,1])~x1 ; the new matrix with the first column being the
  group number
smile=submax(x2,3) ; the 3rd column is the time
```

```
smile=smile[,2]~smile[,4]
smile=sort(smile)

bspl = createfdbasis ("bspline", #(min(smile[,1]), max(smile[,1])),rows
  (smile)+2,smile[,1])

cvl = bfacv(smile[,2], smile[,1], bspl,2,1,lam)
          endp

lmin= nmbrent("minil",10^9,10^10,10^11 )
lmin
```

spdmmo2

Preprocess the data, group the IV and moneyness by strike and time, as is introdunced in 5.5. Plot the IV smile, the first derivative of the estimated volatility with respect to strike together with its no-arbitrage bound, and SPD curve. λ is chosen by GCV.

```
library("fda")
library("finance")
 x=read("rawdata030225.DAT")        ; read the data
t=x[,9]
x=x[,1:6]
x=x~t
x=paf(x,x[,6]==1)
x=paf(x,x[,4]>0.14&& x[,4]<0.15)
t=x[,7]
tau=x[,4]
s=x[,1]
r=x[,3]
k=x[,2]
f=s.*exp(r.*tau)
m=k./f   ; moneyness
y=ImplVola(x[,1:6])
x1=k~t~m~y
x1=x1[1:7|9:rows(x1)] ;delete outlier
int=round(x1[,2]./1800)  ; create the index of time interval for every
  30 minutes
x1=x1[,1]~int~x1[,3:4]
proc(ct)=count(x)
;count number of different groups divided by different strike and time
  index
i=1
ct=matrix(rows(x))
while (i< rows(x))
if ( x[i,1]== x[i+1,1] &&  x[i,2]== x[i+1,2])
ct[i+1]=ct[i]
else
ct[i+1]=ct[i]+1
endif
i=i+1
endo
```

```
endp
x2=count(x1)~x1[,3:4]
proc(subm)=submedian(x)
;claculate the submedian of x with respect to first column of x.
ct=max(x[,1]) ;count the number of different values in first column of
  x
subm=matrix(ct,2)       ;matrix of submedian
i=1
k=1
while (i<=max(x[,1]) )
xx=paf(x,x[,1]==i)
subm[k,1]=median(xx[,2])
subm[k,2]=median(xx[,3])
i=i+1
k=k+1
endo
endp
smile=submedian(x2)
smile=sort(smile)
ix=grid(1,5,24)
ix=ix|rows(smile)
m2=index(smile[,1],ix)
bspl = createfdbasis ("bspline", #(min(m2), max(m2)), 29, m2)
  fdo = data2fd (smile[,2], smile[,1], bspl, 2 , 1, 0.0014915 )
p=grfd(fdo)
eva = evalfd ( smile[,1], fdo, 0)
 eva1 = evalfd ( smile[,1], fdo,1) ; the first derivative of the
   volatility
 eva2 = evalfd ( smile[,1], fdo, 2) ; the second derivative of the
   volatility
lpspd=spdcalm(smile[,1], eva, eva1, eva2, median(s) , median(r), median
  (tau))
spd=(smile[,1].*median(f))~lpspd
li=grid(0,0,rows(spd))
li=spd[,1]~li
li=setmask(li, "line","red","dashed")
der1=smile[,1]~eva1
st=sqrt(mean(tau))
ert=exp(mean(r)*mean(tau))
  rt=mean(r)*mean(tau)
 d1=(log(1./(smile[,1].*ert))+mean(tau)*(mean(r)+0.5.*(eva.^2)))./(eva
   .*st)
  d2=d1-eva.*st
 lb=-cdfn(-d1)./ (st*smile[,1].* pdfn(d1)) ;lower bound of first
   derivative
 ub=cdfn(d2)./ (st*smile[,1].* pdfn(d2));upper bound of first
   derivative
lb=smile[,1]~lb
ub=smile[,1]~ub

;log-normal
mu=mean(log(s))+(mean(r)-mean(y)^2./2).*mean(tau)
```

```
s2=mean(y)^2.*mean(tau)
logn=spd[,1]~(exp(-(log(spd[,1])-mu)^2./(2.*s2))./(sqrt(2.*pi.*s2).*spd
  [,1]))
logn=setmask(logn,"line", "blue","thin")
setsize(600, 600)
pic=createdisplay(2,2)
setmaskp(smile,4,8,5)
p=setmask(p,"line")
show(pic,1,1,smile,p)
setgopt(pic,1,1,"xvalue",0|1,"yvalue",0|1,"title","IV smile with
  moneyness,02-25-2003 tau=0.14167","xlabel","moneyness","ylabel","
  Implied Volatility")
res1 = setmask(spd, "line","thick")
show(pic,1,2,res1,li,logn)
title="SPD:02-25-2003 tau=0.14167"
xlabel="stock price"
ylabel="density"
setgopt(pic,1,2,"title",title,"xlabel",xlabel,"ylabel",ylabel)
setgopt(pic,1,2, "xvalue",0|1)
der1= setmask(der1, "line")
lb=setmask(lb,"line","red","dashed")
ub=setmask(ub,"line","red","dashed")
show(pic,2,2,der1,lb,ub)
show(pic,2,1,der1)
setgopt(pic,2,2,"xvalue",0|1,"title","first derivative and no arbitrage
   bound","xlabel","moneyness")
setgopt(pic,2,1,"xvalue",0|1,"title","first derivative","xlabel","
  moneyness")
setgopt(pic,1,1, "border",0)
setgopt(pic,2,1, "border",0)
setgopt(pic,1,2, "border",0)
setgopt(pic,2,2, "border",0)
```

7.2 Proofs

7.2.1 Convexity of Call Price Function

Proof: See Franke et al. (2003)

[for call option]

Let $\lambda \in [0,1]$ and $K_1 < K_0$. We construct following portfolios A and B at time t:

B 1 a long position in λ calls with exercise price K_1

B 2 a long position in $1-\lambda$ calls with exercise price K_0

A a short position in 1 call with exercise price $K_\lambda := \lambda K_1 + (1-\lambda)K_0$

	value at time t'			
position	$S_{t'} \leq K_1$	$K_1 \leq S_{t'} \leq K_\lambda$	$K_\lambda \leq S_{t'} \leq K_0$	$K_0 \leq S_{t'}$
B1	0	$\lambda(S_{t'} - K_1)$	$\lambda(S_{t'} - K_1)$	$\lambda(S_{t'} - K_1)$
B2	0	0	0	$(1-\lambda)(S_{t'} - K_0)$
A	0	0	$-S_{t'} - K_\lambda$	$-S_{t'} - K_\lambda$
sum	0	$\lambda(S_{t'} - K_1)$	$(1-\lambda)(K_0 - S_{t'})$	0

Thus we can conclude that $\lambda(S_T - K_1) \geq 0$ and $(1-\lambda)(K_0 - S_T) \geq 0$

Hence, the difference of the portfolio values B and A in t has to be non negative, implying:

$$\lambda C_{K_1,T}(S_t,\tau) + (1-\lambda)C_{K_0,T}(S_t,\tau) - C_{K_\lambda,T}(S_t,\tau) \geq 0$$

7.2.2 No Arbitrage Bounds of IV Smile

Proof: See Fengler (2004)

If $K_1 < K_2$ for any expiray date T, we have

$$C_t(K_1,T) \geq C_t(K_2,T), P_t(K_1,T) \leq P_t(K_2,T) \tag{7.1}$$

This can be improved to:

$$C_t(K_1,T) \geq C_t(K_2,T), \frac{P_t(K_1,T)}{K_1} \leq \frac{P_t(K_2,T)}{K_2} \tag{7.2}$$

With the assumption that volatility is a function of strikes, we obtain by differentiating:

$$\frac{\partial C_t}{\partial K} = \frac{\partial C_t^{BS}}{\partial K} + \frac{\partial C_t^{BS}}{\partial \hat{\sigma}} \frac{\partial \hat{\sigma}}{\partial K} \leq 0 \tag{7.3}$$

which implies

$$\frac{\partial \hat{\sigma}}{\partial K} \leq \frac{\partial C_t^{BS}/\partial K}{\partial C_t^{BS}/\partial \hat{\sigma}} \tag{7.4}$$

Differentiating P_t^{BS}/K with respect to K, yields for the lower bound:

$$\frac{\partial \hat{\sigma}}{\partial K} \geq \frac{P_t^{BS}/K - \partial P_t^{BS}/\partial K}{\partial P_t^{BS}/\partial \hat{\sigma}} \tag{7.5}$$

Finally, insert the analytica expression of the option derivatives and the put price and make use of the relationship:

$$e^{-r\tau} K \varphi(d_2) = e^{-\delta\tau} S_t \varphi(d_1) \tag{7.6}$$

we can obtain 2.23 and 2.24

Bibliography

Aït-Sahalia, Y. and Luo, A. (1998). Nonparametric estimation of state-price densities implicit in financial asset prices, *Journal of Finance* **53**: 499–548.

Ané, T. and German, H. (1999). Stochastic volatility and transaction time: an activity-based volatility estimator, *Journal of Risk* **2(1)**: 57–69.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities, *Journal of Political Economy* **81**: 637–654.

Cont, R., Fonseca, J. D. and Durrleman, V. (2002). The dynamics of implied volatility surfaces, *Quantitative Finance* **2(1)**: 45–602.

de Boor, C. (1978). *A Practical Guide to Splines*, Springer, New York.

Drescher, D. (2003). Die dynamik implizierter risikoneutraler dichtefunktionen, *Diplomarbeit,Humboldt-Universität zu Berlin.*

Eubank, R. (1988). *Spline Smoothing and Nonparametric Regression*, Marcel Dekker, New York.

Fan, J. and Gijbels, I. (1996). *Local Polynomial Modelling and Its Applications*, Chapman and Hall, London.

Fengler, M. R. (2004). Semiparametric modelling of implied volatility, *Ph.D Thesis,Humboldt-Universiät zu Berlin.*

Franke, J., Härdle, W. and Hafner, C. (2003). *Introduction to Statistics of Financial Markets*, Springer.

Green, P. and Silverman, B. (1994). *Nonparametric Regression and Generalized Linear Models. A Roughness penalty approach*, Chapmann and Hall, London.

Hafner, R. and Wallmeier, M. (2001). The dynamics of dax implied volatilities, *International Quarterly Journal of Finance* **1(1)**: 1–27.

Härdle, W. and Hlávka, Z. (2005). Dynamics of state price density, *SFB 649 discussion paper 2005-021.*

Härdle, W., Kleinow, T. and Stahl, G. (2002). *Applied Quantitative Finance*, Springer.

Härdle, W., Klinke, S. and Müller, M. (2000). *XploRe Learning Guide*, Springer-Verlag Berlin Heidelberg, New York.

Härdle, W., Müller, M., Sperlich, S. and Werwatz, A. (2004). *Nonparametric and Semiparametric Models*, Springer.

Hastie, T., Tibshirani, R. and Jerome, F. (2001). *The Elements in Statistical Learning*, Springer.

Heckman, N. E. and Ramsay, J. (2000). Penalized regression with model-bases penalties, *The Canadian Journal of Statistics* **28**: 241–258.

Ramsay, J. (1997). Functional data analysis, *C.B. Read and D.L. Banks (Eds.), Encyclopedia of Statistical Sciences, Wiley, New York.*

Ramsay, J. and Dalzell, C. (1991). Some tools for functional data analysis, *Journal of Royal Statistical Society* **B53**: 539–572.

Ramsay, J. and Silverman, B. (1997). *Functional Data analysis*, Springer, New York.

Ramsay, J. and Silverman, B. (2002). *Applied Functional Data analysis. Methods and Case Studies*, Springer, New York.

Rookey, C. (1997). Fully exploiting the information content of intra-day option quotes: Applications in option pricing and risk management, *Technical Report, Department of Finance, University of Arizona.*

Rosenberg, J. (2000). Implied volatility functions: A reprise, *Journal of Derivatives* **7**: 51–64.

Shimko, D. (1993). Bounds of probability, *Risk* **6(4)**: 33–37.

Tompkins, R. (1999). Implied volatility surfaces: Uncovering regularities for options on financial futures, *Working Paper, Vienna University of Technology.*

Ulbricht, J. (2004). Representing functional data as smooth function, *Master Thesis,Humboldt-Universiät zu Berlin.*

Wahba, G. (1990). *Spline models for observational data*, Society for Industrial and Applied Mathematics, Philadelphia.