Estimation of the risk-neutral density function from option prices

Sen Zhou

Iowa State University
Estimation of the risk-neutral density function from option prices

by

Sen Zhou

A dissertation submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Major: Applied Mathematics

Program of Study Committee:
Steven L. Hou, Major Professor
Alexander Roitershtein
Ananda Weerasinghe
Huaiqing Wu
Zhijun Wu

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University
Ames, Iowa
2018

Copyright © Sen Zhou, 2018. All rights reserved.
DEDICATION

I would like to dedicate this dissertation to my family members, Li Ding, Xin Zhou, Chunfang Ji and Jinyu Zhou, without whose support I would not have been able to complete the work.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vi</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>x</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>xi</td>
</tr>
<tr>
<td>CHAPTER 1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Statement of the Problem</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Literature Review</td>
<td>3</td>
</tr>
<tr>
<td>1.2.1 Parametric Methods</td>
<td>3</td>
</tr>
<tr>
<td>1.2.2 Non-parametric Methods</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Dissertation Contributions and Outline</td>
<td>5</td>
</tr>
<tr>
<td>CHAPTER 2. PRELIMINARIES</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Risk-neutral Density Function (RND)</td>
<td>9</td>
</tr>
<tr>
<td>2.1.1 Option prices and the RND</td>
<td>9</td>
</tr>
<tr>
<td>2.1.2 Constraints of the RND and No-arbitrage Opportunities</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Smoothed Implied Volatility Smile Method (SML)</td>
<td>13</td>
</tr>
<tr>
<td>2.3 Support Vector Regression (SVR)</td>
<td>15</td>
</tr>
<tr>
<td>2.3.1 Introduction of SVR</td>
<td>15</td>
</tr>
<tr>
<td>2.3.2 Kernel Tricks</td>
<td>20</td>
</tr>
<tr>
<td>2.3.3 Variations of SVR</td>
<td>23</td>
</tr>
<tr>
<td>2.4 Semi-infinite Programming and Cutting Plane Method</td>
<td>26</td>
</tr>
</tbody>
</table>


LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1</td>
<td>S&amp;P 500 index option market information</td>
<td>37</td>
</tr>
<tr>
<td>Table 3.2</td>
<td>The fitted parameters of S&amp;P 500 index option data by three log-normal density functions</td>
<td>38</td>
</tr>
<tr>
<td>Table 3.3</td>
<td>The RIMSEs, RISBs and RIVs of SML and LPSVR</td>
<td>41</td>
</tr>
<tr>
<td>Table 3.4</td>
<td>The RIMSEs, RISBs and RIVs of SML, LPSVR and $\varepsilon_i$-LPSVR</td>
<td>46</td>
</tr>
<tr>
<td>Table 4.1</td>
<td>The RIMSEs, RISBs and RIVs of LPSVR, $\varepsilon_i$-LPSVR and QPSVR</td>
<td>69</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>The RIMSEs, RISBs and RIVs of LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR</td>
<td>73</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2</td>
<td>Representative S&amp;P 500 volatility curve before and after 1987.</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>The ε-insensitive loss function.</td>
<td>16</td>
</tr>
<tr>
<td>2.4</td>
<td>A nonlinear classification example.</td>
<td>20</td>
</tr>
<tr>
<td>3.1</td>
<td>The fitting result of S&amp;P 500 index option data by three log-normal density functions for Case 1.</td>
<td>47</td>
</tr>
<tr>
<td>3.2</td>
<td>The fitting result of S&amp;P 500 index option data by three log-normal density functions for Case 2.</td>
<td>47</td>
</tr>
<tr>
<td>3.3</td>
<td>The fitting result of S&amp;P 500 index option data by three log-normal density functions for Case 3.</td>
<td>48</td>
</tr>
<tr>
<td>3.4</td>
<td>The fitting result of S&amp;P 500 index option data by three log-normal density functions for Case 4.</td>
<td>48</td>
</tr>
<tr>
<td>3.5</td>
<td>Estimated RND of S&amp;P 500 index option data by SML (left) and LPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.</td>
<td>49</td>
</tr>
<tr>
<td>3.6</td>
<td>Estimated RND of S&amp;P 500 index option data by SML (left) and LPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.</td>
<td>49</td>
</tr>
<tr>
<td>3.7</td>
<td>Estimated RND of S&amp;P 500 index option data by SML (left) and LPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.</td>
<td>50</td>
</tr>
</tbody>
</table>
Figure 3.8 Estimated RND of S&P 500 index option data by SML (left) and LPSVR (right) for Case 4. 'Real’ is the constructed underlying real RND. 'Fit’ is the mean of the estimated 1000 RNDs by SML or LPSVR.

Figure 3.9 The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 1.

Figure 3.10 The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 2.

Figure 3.11 The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 3.

Figure 3.12 The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 4.

Figure 3.13 Estimated RND of S&P 500 index option data by LPSVR (left) and ε_i-LPSVR (right) for Case 1. 'Real’ is the constructed underlying real RND. 'Fit’ is the mean of the estimated 1000 RNDs by LPSVR or ε_i-LPSVR.

Figure 3.14 Estimated RND of S&P 500 index option data by LPSVR (left) and ε_i-LPSVR (right) for Case 2. 'Real’ is the constructed underlying real RND. 'Fit’ is the mean of the estimated 1000 RNDs by LPSVR or ε_i-LPSVR.

Figure 3.15 Estimated RND of S&P 500 index option data by LPSVR (left) and ε_i-LPSVR (right) for Case 3. 'Real’ is the constructed underlying real RND. 'Fit’ is the mean of the estimated 1000 RNDs by LPSVR or ε_i-LPSVR.

Figure 3.16 Estimated RND of S&P 500 index option data by LPSVR (left) and ε_i-LPSVR (right) for Case 4. 'Real’ is the constructed underlying real RND. 'Fit’ is the mean of the estimated 1000 RNDs by LPSVR or ε_i-LPSVR.

Figure 3.17 The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and ε_i-LPSVR for Case 1.

Figure 3.18 The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and ε_i-LPSVR for Case 2.
Figure 3.19 The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 3. ................................................. 56

Figure 3.20 The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 4. ................................................. 56

Figure 4.1 Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR. ... 74

Figure 4.2 Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR. ... 74

Figure 4.3 Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR. ... 75

Figure 4.4 Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR. ... 75

Figure 4.5 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 1. ................................................. 76

Figure 4.6 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 2. ................................................. 76

Figure 4.7 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 3. ................................................. 77

Figure 4.8 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 4. ................................................. 77

Figure 4.9 Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR. ... 78
Figure 4.10 Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR. . . 78

Figure 4.11 Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR. . . 79

Figure 4.12 Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR. . . 79

Figure 4.13 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 1. .......................................................... 80

Figure 4.14 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 2. ......................... 80

Figure 4.15 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 3. ......................... 81

Figure 4.16 The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 4. ......................... 81
ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of this research and the completion of my graduate education.

First and foremost, Dr. Steven L. Hou for his guidance, patience and support throughout this research and the writing of this dissertation. His inspirations and instructions have guided me through my graduate education. At the same time, he has taught me valuable knowledge not only about mathematics but also about career and life. He always encourages me to find what I love to do.

I would also like to thank my committee members for their efforts and valuable suggestions that helped improve this work. Dr. Alexander Roitershtein - I enjoyed taking his advanced stochastic process class. Dr. Ananda Weerasinghe - his work on the blackboard was always fascinating. Dr. Huaiqing Wu - I learned a lot from him about statistics and finance to support my dissertation work. Dr. Zhijun Wu - his numerical and optimization classes helped me build relevant computational and quantitative background.

Last but nor least, I have had a happy time in Ames and have met so many good friends in this peaceful town.
The risk-neutral density function (RND) is a fundamental concept in mathematical finance and is heavily used in the pricing of financial derivatives. The estimation of a well-behaved RND is an ill-posed problem and remains to be a mathematical and computational challenge due to the limitations of data and complicated constraints. Both parametric and non-parametric methods for estimating the RND from option prices have been developed and used in the literature and industry. In this dissertation we propose and study more effective non-parametric methods.

We develop the methods under the framework of linear programming and quadratic programming in combination with Support Vector Regression (SVR).

Under the framework of linear programming, we propose two methods with different penalty schemes. i) The first one named LPSVR uses a general kernel, the log-logistic function, with the standard $\varepsilon$-insensitive loss function to formulate the estimation process into a semi-infinite linear programming optimization problem. We prove the solution of this optimization problem is global by the Cutting Plane Method (CPM). Monte-Carlo simulations are conducted to evaluate the performance of LPSVR. Compared to the benchmark method SML, LPSVR improves both the accuracy and stability. ii) The second one named $\varepsilon_i$-LPSVR modifies LPSVR with the $\varepsilon_i$-insensitive loss function and also formulates the estimation process into a semi-infinite linear programming optimization problem. We may similarly prove the globalness of the solution by CPM. Monte-Carlo simulations are also conducted. Compared to LPSVR, $\varepsilon_i$-LPSVR maintains the stability level and improves the accuracy level by the modified penalty scheme. Overall $\varepsilon_i$-LPSVR outperforms LPSVR.

Under the framework of quadratic programming, we also propose two methods with different penalty schemes. i) The first one named QPSVR uses the RBF kernel with the $\varepsilon$-insensitive square loss function to formulate the estimation process into a semi-infinite quadratic programming
optimization problem. We prove the solution of this optimization problem is global by CPM. Moreover, we prove uniqueness of the solution by the approximation theory. Simulations show that QPSVR maintains the accuracy level as LPSVR. Compared to LPSVR and $\varepsilon_i$-LPSVR, QPSVR improves the stability level by the uniqueness of the solution. Overall QPSVR outperforms LPSVR and $\varepsilon_i$-LPSVR. ii) The second one named $\varepsilon_i$-QPSVR modifies QPSVR with the $\varepsilon_i$-insensitive square loss function and also formulates the estimation process into a semi-infinite quadratic programming optimization problem. We may similarly prove the globalness and uniqueness of the solution by this scheme. Simulations show that $\varepsilon_i$-QPSVR improves both accuracy and stability over LPSVR. Compared to $\varepsilon_i$-LPSVR, $\varepsilon_i$-QPSVR maintains the accuracy level and improves the stability level by the uniqueness of the solution. Compared to QPSVR, $\varepsilon_i$-QPSVR maintains the stability level and improves the accuracy level by the modified penalty scheme. Overall $\varepsilon_i$-QPSVR outperforms LPSVR, $\varepsilon_i$-LPSVR and QPSVR.
CHAPTER 1. INTRODUCTION

1.1 Statement of the Problem

The risk-neutral density function (RND) is a fundamental concept in mathematical finance and is heavily used in pricing financial derivatives. We give the definition as follows:

Definition 1 (RND). The risk-neutral density function for an underlying security is a probability density function for which the current price of the security is equal to the discounted expectation of its future prices.

Under no-arbitrage assumption the RND is guaranteed to exist by asset pricing theory Duffie (2010). In most cases, the number of possible future prices of a security is much larger than the number of its observed prices. This makes the estimation of the RND an underdetermined problem for which there would not be a unique solution unless some additional constraints are imposed Monteiro et al. (2008).

Specifically, we consider the application of the RND in option markets. Option markets are believed to contain rich information about market participants’ expectations through its implied RND. As an illustration, we introduce a simple, idealized example below. Assume there is a call option on an underlying asset with the risk-free rate being zero. The value of the option will have a 50% probability of being $10 and a 50% probability of being $0 at expiration. A reasonable price today of the call option should be: $10 \times 0.5 + 0 \times 0.5 = 5$ dollars. Conversely if we know the option price today, we can infer the underlying asset’s probability corresponding to its future value. The probability

\[
 f = \begin{cases} 
 0.5, \text{ if option value is } $10 \\
 0.5, \text{ if option value is } $0 
\end{cases}
\]

here is the RND we are discussing about.
In the risk-neutral valuation approach Cox and Ross (1976), the price of a European call option of a stock is expressed as follows:

\[
C(K) = e^{-rt} \int_0^\infty \max(0, S - K) f(S) dS = e^{-rt} \int_K^\infty (S - K) f(S) dS
\]  

where \( f(\cdot) \) is the RND, \( K \) is the strike price, \( S \) is the stock price at maturity, \( t \) is the time to maturity, \( r \) is the risk-free rate.

Equation (1.1.1) can be interpreted as follows: the call option price today is equal to its discounted expectation at expiration with the value \( \max(0, S - K) \) and the corresponding probability \( f(S) \). The discounting factor is \( e^{-rt} \). Our goal is to estimate the RND \( f(\cdot) \) from option prices in the market.

Since late 1980s, with the availability of powerful computers and option database, financial institutions have paid growing attention to the estimation of the RND. A variety of techniques have been developed to estimate the RND through the underlying assets’ option prices. However, it is not easy to obtain a well-behaved RND if the following issues are not properly addressed.

First, non-uniqueness of the RND. The asset pricing theory guaranteed the existence of the RND. However, it is not unique because different probability density function may produce the same expectation.

Second, limitations of data. We only have option prices at discrete and limited strike prices in the market, while the RND is supported on \( [0, \infty) \).

Third, market noises. Market noises include spreads of bid-ask prices, non-synchronous trading and other frictions in the market.

Fourth, no-arbitrage opportunities. No arbitrage opportunities impose additional restrictions on the RND which complicate the estimation process.

Last but not least, constraints of the RND. The RND is a probability density function that must be non-negative and integrate to one.

Despite of the aforementioned difficulties in estimating the RND, various techniques have been produced and implemented due to the practical importance of the RND. Those techniques generate
inconsistent solutions, and their pros and cons have been discussed in the literature. To the best of our knowledge, no consensus has been reached as to the choice of a best technique.

1.2 Literature Review

The existing methods for the estimation of the RND can be categorized as: Parametric methods and Non-parametric methods Jackwerth (1999),Jackwerth (2004).

1.2.1 Parametric Methods

Generally, parametric methods start with a distribution involving a set of parameters and make adjustments to the assumed distribution. Then based on the distribution one can price the options and determine the parameters of the distribution by minimizing the pricing error.

The most classical approach in this category is modeling the RND as a two-parameter lognormal distribution with unknown mean and volatility on which the Black-Scholes model is based. However, this method has been proven to be not flexible enough to match the option prices in the real world.

Approaches within parametric methods are diverse. There are roughly three groups: expansion methods, generalized distribution methods, and mixture methods Jackwerth (2004).

Specifically, the expansion methods begin with a simple distribution (e.g. log-normal) and then add high-level correction terms to get additional flexibility Corrado and Su (1996),Jarrow and Rudd (1982),Rompolis and Tzavalis (2008),Rubinstein (1998). The generalized distribution methods use more flexible and complex distributions. Rather than the typical two parameters for the mean and the volatility, skewness and kurtosis parameters are often added in this method Corrado (2001),Fabozzi et al. (2009),Sherrick et al. (1996). The mixture methods combine several simple distributions together as weighted average to increase flexibility Giacomini et al. (2008),Melick and Thomas (1997),Ritchey (1990).

Both pros and cons of parametric methods have been well discussed in the literature Jackwerth (1999),Jackwerth (2004). On one hand, parametric methods have advantages in that they only involve a few parameters so that the computation process is not heavy. However, on the other
hand, they have drawbacks if an inappropriate process is assumed or a distribution that is not flexible enough to fit the data is picked.

The adjustments made to the distribution to gain flexibility also have several limitations. For example, for the expansion methods, the added correction terms are not guaranteed to preserve the constraints of a probability density function. For the mixture methods, the number of parameters increases rapidly with more combined distributions. Moreover, mixture methods tend to over-fit the data and the obtained RND tends to have sharp spikes Giamouridis and Tamvakis (2002).

1.2.2 Non-parametric Methods

The second category is non-parametric methods. Non-parametric methods, rather than assuming a probability distribution, use more general functions to achieve greater flexibility in fitting option prices using certain criteria.

One well-known approach within this category is the smoothed implied volatility smile method (SML). SML utilizes the fact that the second derivative of the option price function is proportional to the RND. It has different modifications and is very easy to implement. But it cannot guarantee non-negativity due to the involvement of second derivatives.

Approaches within the non-parametric methods category are diverse and can be roughly divided into three main groups: maximum entropy methods, kernel regression methods, curve fitting methods Jackwerth (2004).

The maximum entropy methods first pick a prior distribution, which is always a log-normal distribution. Then they find the posterior RND by fitting the option prices and presume the least information relative to the prior probability distribution Buchen and Kelly (1996), Rompolis (2010), Stutzer (1996). The kernel methods are based on the idea that each data point could be viewed as the center of a region where the true function passes. The farther an estimated point is away from the observed data point, the less likely the true function passes through that point. A kernel $K(x)$, which is often assumed to be the normal destiny function, is picked to measure the corresponding drop in the likelihood when a function moves away from the data point. The kernel
methods construct the RND locally based on the selected kernel Aït-Sahalia and Duarte (2003), Aït-Sahalia and Lo (1998), Li and Zhao (2009). The curve fitting methods use flexible functions such as polynomials or splines to fit the option prices or implied volatilities and then convert them to the RND. This method could also fit the RND directly Campa et al. (1998), Du et al. (2012), Jackwerth and Rubinstein (1996), Monteiro et al. (2008), Rubinstein (1994).


The pros of non-parametric methods are that they do not need to assume a distribution for the RND and thus avoid the possibility of choosing an inappropriate one. They also allow more flexibility in the RND. The cons of non-parametric are that there are usually more parameters involved thus more computation. Other cons differ by the methods. For example, the maximum entropy methods are noise sensitive. They use the logarithm of the ratios of probabilities, which can go to large negative values. The kernel methods construct the RND locally and do not work well for data with a lot of gaps. The curve fitting methods could get a good interpolation results within the strikes. Extrapolation beyond the available data does not have a consensus at this point and needs to be addressed separately Jackwerth (1999), Jackwerth (2004).

1.3 Dissertation Contributions and Outline

Based on the current situation, we search for more practical and flexible methods to estimate the RND which would also satisfy all the constraints. Statistical learning methods have drawn our attention.

Support Vector Regression (SVR) belongs to the statistical learning methods which allows users to control the complexity of the function and the goodness of fit of the data. Researchers have applied SVR in finance area to estimate the volatilities, interest rates, option prices and so on Andreou et al. (2009), d’Almeida Monteiro (2010), Pérez-Cruz et al. (2003).
Ian used SVR to estimate the implied volatility function and converted it back to option prices Ian and Choo (nd). Andreou simply applied SVR to estimate the European option prices to gain additional flexibility Andreou et al. (2009). Feng adopted the idea of SVR and used a loss function which penalizes each data point with at least a fixed amount to estimate the RND in terms of linear programming. He then used the RND to estimate the risk aversion function Feng and Dang (2016).

Inspired by the aforementioned, we propose our methods based on SVR in terms of both linear programming and quadratic programming. The estimation of the RND is finally formulated into an optimization problem as in Feng and Dang (2016). One thing that we notice is missing in these literature is the discussion about the uniqueness of the solution, especially in terms of SVR, where the objective function is dependent on the chosen kernel function. We prove that our schemes guarantee a global solution both in the linear and the quadratic cases. Moreover we prove that our schemes guarantee a unique solution in the quadratic case. Through the designed Monte-Carlo simulations, we show that the estimation variance can be reduced by the uniqueness of the solution.

Another improvement in this dissertation is about the penalty scheme used in the optimization problem’s objective function. The standard and most used penalty scheme in SVR is the \(\varepsilon\)-insensitive loss function which chooses a common penalty threshold for every data point. We show that our methods with different modified loss functions can keep the advantage of sparsity and improve the estimation accuracy through the designed simulations.

Besides our methods improve others in the following aspects:

- SVR is a flexible method which also preserves the advantage of sparsity.
- SVR is robust to noises by trade-off parameters to control the complexity of the function and the goodness of fit of the data.
- Reduce the bias by using vary tube sizes in the loss function and reduce the variance by guaranteeing the unique estimation of the RND.
- Both no-arbitrage constrains and the RND constraints are satisfied.
• Avoid the "Curse of Differentiation" by modeling the RND directly.

• Our approach belongs to non-parametric methods category which avoids specifying an inappropriate distribution. It will fully recover the RND on the entire support $[0, \infty)$.

Besides the above mentioned improvement, we give a brief discussion of the organization of the dissertation below.

In chapter 2 we review some preliminaries to set up our background knowledge. We talk about the relationship between the RND and option prices. The constraints of the estimation process are also derived. Then we introduce the benchmark, Smoothed Implied Volatility Smile method (SML). A detailed discussion of Support Vector Regression (SVR) is given and we review the semi-infinite programming and the corresponding Cutting Plane Method (CPM) which we use later as a tool to solve our optimization problem.

In chapter 3 we propose the non-parametric methods based on SVR in terms of linear programming. First we develop a method named LPSVR by using the log-logistic kernel and the standard $\varepsilon$-insensitive loss function. The estimation process is formulated as a semi-infinite linear programming problem and the globalness of the solution is proved. Then we describe the implementation steps of the benchmark method SML followed by a detailed explanation of our designed Monte-Carlo simulations. The comparison results between SML and LPSVR are given and the measurements show that LPSVR outperforms SML. We develop another method named $\varepsilon_i$-LPSVR by changing the penalty scheme to the $\varepsilon_i$-insensitive loss function, i.e., different penalty thresholds for different data points. This would increase a big amount of calculation if we have to arbitrarily search for the thresholds of all data points. However based on the specialty of the option price data, the thresholds can be set as a part of the bid-ask spreads which perfectly solved this issue. Simulations show that $\varepsilon_i$-LPSVR improves the accuracy level compared to LPSVR by this modified penalty scheme.

In chapter 4 we propose the non-parametric methods based on SVR in terms of quadratic programming. First we develop a method named QPSVR by using the RBF kernel and the square of the standard $\varepsilon$-insensitive loss function. The estimation process is formulated as a semi-infinite
quadratic programming problem and the globalness and uniqueness of the solution are proved. We take LPSVR and $\varepsilon_i$-LPSVR as benchmarks. Followed by the same Monte-Carlo simulations in section 3.3, the comparison results show that QPSVR improves both LPSVR and $\varepsilon_i$-LPSVR in the stability level by the uniqueness of the solution. QPSVR maintains the accuracy level as LPSVR and performs worse than $\varepsilon_i$-LPSVR. In total, QPSVR outperforms LPSVR and $\varepsilon_i$-LPSVR with smaller RIMSEs. We propose another method named $\varepsilon_i$-QPSVR by changing the penalty scheme to the square of the $\varepsilon_i$-insensitive loss function, i.e., different penalty thresholds inside the square for different data points. Simulations show that $\varepsilon_i$-QPSVR improves the accuracy level by the modified penalty scheme and also improves the stability level by the uniqueness of the solution. In total $\varepsilon_i$-QPSVR has the best performance among all four proposed methods.

In chapter 5 we summarize the main contents in the dissertation and discuss some future research potentials.
CHAPTER 2. PRELIMINARIES

In this chapter we introduce some background knowledge of the estimation problem. Section 2.1 discusses the relationship between option prices and the RND. It also summarizes the constraints of the estimation problem. Section 2.2 reviews a famous non-parametric method for estimating the RND, the Smoothed Implied Volatility Smile method (SML) - which will be used as a benchmark later to compare with our proposed non-parametric methods. Section 2.3 introduces the idea of the Support Vector Regression (SVR) which is the base of our method. Kernel tricks and variations of SVR are also reviewed. In section 2.4 we review the semi-infinite programming and the corresponding Cutting Plane Method (CPM) which we use later as a tool to solve our optimization problem.

2.1 Risk-neutral Density Function (RND)

2.1.1 Option prices and the RND

In the risk-neutral valuation approach Cox and Ross (1976), the price of a European call option of a stock is expressed as follows:

\[ C(K) = e^{-rt} \int_{0}^{\infty} \max(0, S - K) f(S) dS \]

\[ = e^{-rt} \int_{0}^{\infty} (S - K) f(S) dS \tag{2.1.1} \]

where \( f(\cdot) \) is the RND, \( K \) is the strike price, \( S \) is the stock price at maturity, \( t \) is the time to maturity, \( r \) is the risk-free rate.

Differentiate the above equation with respect to the strike price \( K \):

\[ \frac{\partial C(K)}{\partial K} = (S - K) f(S) \bigg|_{S=K} + e^{-rt} \int_{K}^{\infty} \frac{\partial(S - K) f(S)}{\partial K} dS \]

\[ = 0 + e^{-rt} \int_{K}^{\infty} -f(S) dS \]

\[ = -e^{-rt} \int_{K}^{\infty} f(S) dS \tag{2.1.2} \]
Differentiate it with respect to the strike price $K$ again:

$$\frac{\partial^2 C(K)}{\partial K^2} = e^{-rt} f(S) \bigg|_{S=K} = e^{-rt} f(K)$$

So the RND is given as in Breeden and Litzenberger (1978):

$$f(K) = e^{rt} \frac{\partial^2 C(K)}{\partial K^2}$$

(2.1.4)

This relationship implies that if we have the option price function, differentiate it twice with respect to the strike price and multiply by the discounting factor, we will obtain the RND.

### 2.1.2 Constraints of the RND and No-arbitrage Opportunities

In this section, we consider the constraints needed to be imposed on the estimation of the RND. The RND is a probability density function, so it is non-negative and integrates to one. The other constraint is there are no-arbitrage opportunities.

First the RND should be non-negative and integrate to one, i.e.:

$$f(K) \geq 0, \ K \in [0, \infty)$$

$$\int_0^\infty f(S)dS = 1$$

(2.1.5)

Second there should be no-arbitrage opportunities. The constraint of no-arbitrage opportunities on the call option prices is Jackwerth and Rubinstein (1996):

$$\max(0, S_0e^{-\delta t} - Ke^{-rt}) \leq C(K) \leq S_0e^{-\delta t}$$

where $K$ is the strike price, $r$ is the risk-free rate, $\delta$ is the dividend yield rate, $S_0$ is the current stock price, $t$ is the time to maturity. To discuss this constraint, we will derive the information we already have related to option prices based on the risk-neutral valuation approach.

Recall equation (2.1.1), (2.1.2), (2.1.3):

$$C(K) = e^{-rt} \int_K^\infty (S - K)f(S)dS$$

$$C'(K) = \frac{\partial C(K)}{\partial K} = -e^{-rt} \int_K^\infty f(S)dS$$
\[ C''(K) = \frac{\partial^2 C(K)}{\partial K^2} = e^{-rt} f(K) \]

Notice \( f(K) \geq 0, \int_K^\infty f(S)dS \geq 0 \), so:

\[ C'(K) \leq 0, \ C''(K) \geq 0, \ K \in [0, \infty) \]

Notice that \( f(K) \) cannot be identically 0 on \([0, \infty)\), so do \( C'(K) \) and \( C''(K) \). Then \( C'(K) \) will be an increasing function, and \( C(K) \) will be a convex decreasing function on their domain.

For \( C'(K) \):

\[ C'(\infty) = -e^{-rt} \int_\infty^\infty f(S)dS = 0 \] (2.1.6)

\[ C'(0) = -e^{-rt} \int_0^\infty f(S)dS = -e^{-rt} \] (2.1.7)

Since \( C'(K) \) is an increasing function on \([0, \infty)\), so:

\[ -e^{-rt} \leq C'(K) \leq 0 \] (2.1.8)

For \( C(K) \):

\[ C(\infty) = e^{-rt} \int_\infty^\infty (S-\infty)f(S)dS = 0 \] (2.1.9)

\[ C(0) = e^{-rt} \int_0^\infty (S-0)f(S)dS = e^{-rt}E(S) = S_0e^{-\delta t} \] (2.1.10)

where \( E(S) \) is the expected value of the stock price at time \( t \). The last equation comes from the martingale property in option pricing theory Ingersoll Jr (1989):

\[ e^{-(r-\delta)t}E(S) = S_0 \]

where \( e^{-(r-\delta)t} \) is the discounting factor. Since \( C(K) \) is a decreasing function on \([0, \infty)\), so:

\[ 0 \leq C(K) \leq S_0e^{-\delta t} \] (2.1.11)
Notice $C'(K)$ is an increasing function, i.e., $C''(K) \geq C'(0)$. Then:

\[
C(K) = C(0) + \int_0^K C'(S) dS \\
\geq C(0) + \int_0^K C'(0) dS \\
= C(0) + KC'(0) \\
= S_0 e^{-\delta t} - Ke^{-rt}
\]

(2.1.12)

So for $C(K)$:

\[
\max(0, S_0 e^{-\delta t} - Ke^{-rt}) \leq C(K) \leq S_0 e^{-\delta t}
\]

(2.1.13)

This is exactly what the constraint of no-arbitrage opportunities on the call option prices is. So it is suffice to have the following conditions to guarantee there are no-arbitrage opportunities:

\[
\begin{aligned}
C''(K) &\geq 0, \quad K \in [0, \infty) \\
C''(0) &= -e^{-rt} \\
C'(\infty) &= 0 \\
C(0) &= S_0 e^{-\delta t} \\
C(\infty) &= 0
\end{aligned}
\]

Recall the constraints of the RND (2.1.5):

\[
f(K) \geq 0, \quad K \in [0, \infty) \\
\int_0^\infty f(S) dS = 1
\]

and equation (2.1.1), (2.1.2), (2.1.3):

\[
C(K) = e^{-rt} \int_K^\infty (S - K) f(S) dS \\
C'(K) = -e^{-rt} \int_K^\infty f(S) dS \\
C''(K) = e^{-rt} f(K)
\]
By the above equation \( C'(\infty) = 0 \) and \( C(\infty) = 0 \) is automatically satisfied. And:

\[
\begin{align*}
  f(K) \geq 0 & \iff C''(K) \geq 0, K \in [0, \infty) \\
  \int_0^\infty f(S)dS = 1 & \iff C'(0) = -e^{-rt}
\end{align*}
\]

(2.1.14)

So in total we have the following constrains to be incorporated into our estimation of the RND:

\[
\left\{
\begin{array}{l}
  f(K) \geq 0, K \in [0, \infty) \\
  \int_0^\infty f(S)dS = 1 \\
  C(0) = S_0 e^{-\delta t}
\end{array}
\right.
\]

(2.1.15)

### 2.2 Smoothed Implied Volatility Smile Method (SML)

Among all the non-parametric methods for estimating the RND, Smoothed Implied Volatility Smile method (SML) is famous for its simplicity of implementation. To further illustrate SML, we will briefly review the concept called the Volatility Smile. The Black-Scholes formula for a European call option is:

\[
C = S_0 e^{-\delta t} N(d_1) - K e^{-rt} N(d_2)
\]

\[
d_1 = \frac{\ln(S_0/K) + (r - \delta + \sigma^2/2)t}{\sigma \sqrt{t}}
\]

\[
d_2 = \frac{\ln(S_0/K) + (r - \delta - \sigma^2/2)t}{\sigma \sqrt{t}}
\]

(2.2.1)

where \( C \) is the European call option price, \( K \) is the strike price, \( t \) is the time to maturity, \( r \) is the risk-free rate, \( \delta \) is the dividend yield rate, \( S_0 \) the current stock price, \( N \) is the standard normal cumulative distribution, \( \sigma \) is the volatility of the stock (standard deviation of the log returns of the stock).

Notice that every parameter in the Black-Scholes formula is known in the market except the volatility \( \sigma \), which is also called the implied volatility. Given the option prices and other information, we can inversely derive the implied volatility \( \sigma \). One of the assumptions in the Black-Scholes formula is that the implied volatility \( \sigma \) should be independent of the strike price \( K \). This pattern seems
to be true before 1987. But it does not hold any more after the 1987’s market crash and shows a curve pattern which is called the Volatility Smile as depicted in Figure 2.2\(^1\).

![Figure 2.2: Representative S&P 500 volatility curve before and after 1987.](image)

Recall (2.1.4) we know that:

\[
f(K) = e^{rt} \frac{\partial^2 C(K)}{\partial K^2}
\]

The SML explicitly utilizes this result.

The main idea of SML is summarized as follows:

- Convert the available call option prices in the market to implied volatilities using the Black-Scholes formula.
- Fit the implied volatilities by certain criteria.
- Use Black-Scholes formula again to convert the fitted implied volatilities back to an option price function.
- Compute the second derivative of the option price function to estimate the RND.

Researchers notice that the implied volatilities curve are much more smoother than the option prices curve itself. And that is why they choose to model the implied volatilities to get back to the option price function instead of modeling the option prices directly.

Different techniques have been raised to fit the implied volatilities. Shimko proposed to use a simple quadratic polynomial to fit the volatility against the strike price within the available data points Shimko (1993). He used lognormal tails outside the available strikes. Malz modified Shimko’s method by fitting the implied volatility against the option delta \( \delta = \frac{\partial C}{\partial S} \) rather than the strike price \( K \) Malz (1997). He argued that it is more accurate to fit the implied volatility against the option delta rather than the strike. Campa, Chang and Reider proposed to use natural spline, rather than low-order polynomials to fit the implied volatility against the strike price Campa et al. (1998). Through the natural spline they could control the smoothness of the fitted function and add flexibility to the model. Bliss and Panigirtzoglou followed Malz and Campa Bliss and Panigirtzoglou (2002). They proposed to use a smoothing cubic spline to fit the implied volatility against the option delta.

Here we choose to fit the implied volatilities against the strike price by a smoothing cubic spline as in Campa et al. (1998). And we are going to use cross-validation to choose the smoothing parameter.

### 2.3 Support Vector Regression (SVR)

In this section we introduce the idea of Support Vector Regression (SVR) which forms the base of our estimation methods.

#### 2.3.1 Introduction of SVR

Suppose we have a data set \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset X \times R \), where \( X = R^d \). In SVR, the goal is to find a function \( f(x) \) that best approximates these data points and also as flat as possible.

We begin with the case of a linear function \( f \), taking the form:

\[
    f(x) = \langle w, x \rangle + b \tag{2.3.1}
\]

where \( w \in X \), and \( b \in R \), \( \langle \cdot, \cdot \rangle \) denotes the dot product in \( X \). Flatness in the case of equation (2.3.1) means a small \( w \). One way to ensure this is to minimize the norm, i.e., \( \|w\|^2 = \langle w, w \rangle \).
So the goal is to solve the following problem:

$$\min_{w,b} \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} L(y_i, f(x_i))$$  \hspace{1cm} (2.3.2)$$

Here $L(y_i, f(x_i)$ is the loss function which describes how the function $f(x)$ approximates these data points. $\lambda$ is a positive parameter which determines the trade-off between the flatness of $f$ and the goodness of fit of the data.

There are a variety types of loss functions. The standard and most common used one is the $\varepsilon$-insensitive loss function, which is given by:

$$|\xi_i|_{\varepsilon} := \begin{cases} 
0, & \text{if } |\xi_i| \leq \varepsilon \\
|\xi_i| - \varepsilon, & \text{otherwise} \end{cases}$$

where $\varepsilon \geq 0$. Figure 2.3 explains this situation graphically.

![Figure 2.3: The $\varepsilon$-insensitive loss function.](image)

This loss function only pays attention to the points outside the tube (shaded area) and neglect points within $\varepsilon$ distance to the proposed function. The loss is counted in a linear form, i.e., the distance from the outside points to the closest boundary of the tube. So equation (2.3.2) becomes:

$$\min_{w,b} \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_{\varepsilon}$$  \hspace{1cm} (2.3.3)$$
For a selected $\varepsilon$ and $\lambda$, introducing slack variables $\xi, \xi^*_i$, we can rewrite equation (2.3.3) into a Quadratic Programming (QP) optimization problem as stated in Vapnik (2013):

$$
\min_{w, b, \xi_i, \xi^*_i} \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} (\xi_i + \xi^*_i)
$$

subject to:

$$
y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i, \quad i = 1, \ldots, n.
$$

$$
\langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi^*_i, \quad i = 1, \ldots, n.
$$

$$
\xi_i, \xi^*_i \geq 0, \quad i = 1, \ldots, n.
$$

where $w, b, \xi_i, \xi^*_i$ are the variables of the problem.

Next we will discuss the dual formulation of the optimization problem (2.3.4). Not only because in most cases the dual form provides an easier way to solve the problem, but also it naturally extends the linear function $f$ to nonlinear functions and explains what Support Vector is.

The Lagrange function of the optimization problem (2.3.4) is:

$$
L := \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} (\xi_i + \xi^*_i) - \sum_{i=1}^{n} l_i \xi_i - \sum_{i=1}^{n} l^*_i \xi^*_i 
$$

$$
- \sum_{i=1}^{n} d_i (\varepsilon + \xi_i - y_i + \langle w, x_i \rangle + b)
$$

$$
- \sum_{i=1}^{n} d^*_i (\varepsilon + \xi^*_i + y_i - \langle w, x_i \rangle - b)
$$

(2.3.5)

Here $L$ is the Lagrange function and $l_i, l^*_i, d_i, d^*_i$ are Lagrange multipliers

$$
l_i, l^*_i, d_i, d^*_i \geq 0
$$

(2.3.6)

The dual objective function is:

$$
g(l_i, l^*_i, d_i, d^*_i) = \min_{w, b, \xi_i, \xi^*_i} \quad L
$$

(2.3.7)

Here $w, b, \xi_i, \xi^*_i$ are the primal variables in the optimization problem (2.3.4)

By setting the derivatives of $L$ with respect to primal variables equal to zero we have:

$$
\frac{\partial}{\partial w} L = w - \sum_{i=1}^{n} (d_i - d^*_i) x_i = 0
$$

(2.3.8)
\[
\frac{\partial}{\partial b} L = \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) = 0
\]  
(2.3.9)

\[
\frac{\partial}{\partial \xi_{i}} L = \lambda - d_{i} - l_{i} = 0, \ i = 1, \ldots, n.
\]  
(2.3.10)

\[
\frac{\partial}{\partial \xi_{i}^{*}} L = \lambda - d_{i}^{*} - l_{i}^{*} = 0, \ i = 1, \ldots, n.
\]  
(2.3.11)

i.e.,
\[
w = \sum_{i=1}^{n} (d_{i} - d_{i}^{*}) x_{i}
\]  
(2.3.12)

\[
\sum_{i=1}^{n} (d_{i}^{*} - d_{i}) = 0
\]  
(2.3.13)

\[
l_{i} = \lambda - d_{i}, \ i = 1, \ldots, n.
\]  
(2.3.14)

\[
l_{i}^{*} = \lambda - d_{i}^{*}, \ i = 1, \ldots, n.
\]  
(2.3.15)

Plug equation (2.3.12) - (2.3.15) back to (2.3.5), we have:
\[
L = \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} (\xi_{i} + \xi_{i}^{*}) - \sum_{i=1}^{n} (\lambda - d_{i}) \xi_{i} - \sum_{i=1}^{n} (\lambda - d_{i}^{*}) \xi_{i}^{*}
\]
\[
- \sum_{i=1}^{n} d_{i}(\varepsilon + \xi_{i} - y_{i} + \langle w, x_{i} \rangle + b) - \sum_{i=1}^{n} d_{i}^{*}(\varepsilon + \xi_{i}^{*} + y_{i} - \langle w, x_{i} \rangle - b)
\]
\[
= \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} (d_{i} + d_{i}^{*}) \varepsilon - \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) y_{i} - \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) \langle w, x_{i} \rangle + \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) b
\]
\[
= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{i}^{*} - d_{j}^{*})(d_{j}^{*} - d_{i}) \langle x_{i}, x_{j} \rangle - \sum_{i=1}^{n} (d_{i}^{*} + d_{i}^{*}) \varepsilon - \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) y_{i}
\]
\[
+ \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{i}^{*} - d_{i})(d_{j}^{*} - d_{j}) < x_{i}, x_{j} >
\]
\[
= - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{i}^{*} - d_{j}^{*})(d_{j}^{*} - d_{i}) \langle x_{i}, x_{j} \rangle - \sum_{i=1}^{n} (d_{i}^{*} + d_{i}^{*}) \varepsilon - \sum_{i=1}^{n} (d_{i}^{*} - d_{i}) y_{i}
\]  
(2.3.16)
From equation (2.3.14), (2.3.15) we also have:

\[ l_i = \lambda - d_i \geq 0, \ i = 1, \ldots, n. \]  
\[ l_i^* = \lambda - d_i^* \geq 0, \ i = 1, \ldots, n. \]  

i.e.

\[ \lambda \geq d_i, \ i = 1, \ldots, n. \]  
\[ \lambda \geq d_i^*, \ i = 1, \ldots, n. \]  

Combining equation (2.3.6), (2.3.13), (2.3.16), (2.3.18), we have the Dual problem:

\[
\begin{aligned}
\max_{d_i, d_i^*} & \ - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_i - d_i^*) (d_j - d_j^*) \langle x_i, x_j \rangle \\
& \ - \sum_{i=1}^{n} (d_i + d_i^*) \varepsilon - \sum_{i=1}^{n} (d_i^* - d_i) y_i \\
\text{s.t.} & \ \sum_{i=1}^{n} (d_i^* - d_i) = 0, \ i = 1, \ldots, n. \\
& \ 0 \leq d_i \leq \lambda, \ i = 1, \ldots, n. \\
& \ 0 \leq d_i^* \leq \lambda, \ i = 1, \ldots, n.
\end{aligned}
\]  

This becomes a Quadratic Programming problem in terms of \(d_i, d_i^*, i = 1, \ldots, n,\) the introduced Lagrange variables.

If we found the optimal solutions to the primal and dual problems with duality gap being 0, i.e. the KKT conditions satisfied, then from the dual complementarity condition, which states that at optimal all Lagrange terms disappear, we have:

\[ d_i (\varepsilon + \xi_i - y_i + \langle w, x_i \rangle + b) = 0, \ i = 1, \ldots, n. \]  
\[ d_i^* (\varepsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b) = 0, \ i = 1, \ldots, n. \]  

\[ l_i \xi_i = (\lambda - d_i) \xi_i = 0, \ i = 1, \ldots, n. \]  
\[ l_i^* \xi_i^* = (\lambda - d_i^*) \xi_i^* = 0, \ i = 1, \ldots, n. \]  

For points strictly inside the tube, from the \(\varepsilon-\)insensitive loss function we know that \(\xi_i, \xi_i^* = 0,\) and

\[ \langle w, x_i \rangle + b - y_i < \varepsilon, \ i = 1, \ldots, n. \]  
\[ y_i - \langle w, x_i \rangle - b < \varepsilon, \ i = 1, \ldots, n. \]
So from equation (2.3.20) we must have $d_i, d_i^* = 0$ for points strictly inside the tube.

Recall equation (2.3.12):

$$w = \sum_{i=1}^{n} (d_i - d_i^*) x_i$$

We can see that $w$ is only determined by the points on the boundary and outside of the tube where $d_i - d_i^* \neq 0$. And these points are called Support Vectors.

Our function now becomes:

$$f(x) = \langle w, x \rangle + b = \sum_{i=1}^{n} (d_i - d_i^*) \langle x_i, x \rangle + b$$  \hspace{1cm} (2.3.23)

Notice here we have the inner product term $\langle x_i, x \rangle$ which makes it easy to apply Kernel Tricks and extend linear cases to nonlinear cases.

**2.3.2 Kernel Tricks**

Next we introduce the idea of Kernel Tricks with a simple classification problem.

Figure 2.4 explains this situation graphically.

![Figure 2.4: A nonlinear classification example.](image)
Suppose we have a data set \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \subset X \times \{1, -1\} \), \( X = (z_1, z_2) = R^2 \), as shown in Figure 2.4(a). It is obvious that the best classification curve is an ellipse in the space \( R^2 \) as shown in Figure 2.4(b):

\[
w_1 z_1^2 + w_2 z_2^2 + b = 0
\]

Now let us define a projection \( \phi : (z_1, z_2) \rightarrow (q_1, q_2) \) to map the data to a different feature space:

\[
(q_1, q_2) = \phi(z_1, z_2) = (z_1^2, z_2^2)
\]

So the classification curve becomes a line:

\[
w_1 q_1 + w_2 q_2 + b = 0
\]

This inspires us that by mapping the data to a higher dimensional space, we would have more chance to solve a nonlinear problem in a linear form.

Most of the time, the mapping is not done explicitly because there is a computational cheaper way, i.e. Kernel Tricks.

Recall that in equation (2.3.23) we have an inner product \( \langle x_i, x \rangle \) in function \( f \). Let us consider a projection \( \phi : R^3 \rightarrow R^9 \):

\[
\phi(Z) = \phi(z_1, z_2, z_3) = \begin{bmatrix}
z_1 z_1 \\
z_1 z_2 \\
z_1 z_3 \\
z_2 z_1 \\
z_2 z_2 \\
z_2 z_3 \\
z_3 z_1 \\
z_3 z_2 \\
z_3 z_3
\end{bmatrix}
\] (2.3.24)
The inner product in $\mathbb{R}^9$ can also be written as:

$$\langle \phi(Z), \phi(Y) \rangle = \phi(Z)^T \phi(Y)$$

$$= \sum_{i,j=1}^{3} (z_i z_j)(y_i y_j)$$

$$= \sum_{i=1}^{3} \sum_{j=1}^{3} z_i z_j y_i y_j$$

$$= \sum_{i=1}^{3} (z_i y_i) \sum_{j=1}^{3} (z_j y_j)$$

$$= \sum_{i=1}^{3} (z_i y_i) \sum_{i=1}^{3} (z_i y_i)$$

$$= (Z^T Y)^2$$  \hfill (2.3.25)

So we can define a kernel:

$$\mathcal{K}(Z, Y) := (Z^T Y)^2 = \phi(Z)^T \phi(Y) = \langle \phi(Z), \phi(Y) \rangle$$  \hfill (2.3.26)

Notice that the computation of equation (2.3.24) takes $O(n^2)$ times while the computation of equation (2.3.25) only takes $O(n)$ times where $n$ is the dimension of the input.

If our problem is in terms of an inner product and we are only interested in the inner product in the feature space instead of the mapping $\phi$ itself, we can use this Kernel Trick to simply our computation. Equation (2.3.26) is a linear kernel example.

Next we introduce the standard of kernel functions. We need the following definition and theorem.

**Definition 2** [Kernel Matrix Ng (2008)]. Consider a set of points $\{x_1, \ldots, x_m\}$, where $x_i \in \mathbb{R}^d$. And let a square, $m$-by-$m$ matrix $M$ be defined so that its $(i,j)$-entry $M_{ij} = \mathcal{K}(x_i, x_j)$, where $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is a given function. Then this matrix $M$ is called the Kernel Matrix of the function $\mathcal{K}$.

**Theorem 1** [Mercer Ng (2008)]. Let $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ be given. Then for $\mathcal{K}$ to be a valid (Mercer) kernel, it is necessary and sufficient that for any $\{x_1, \ldots, x_m\}$, where $x_i \in \mathbb{R}^d$, the corresponding Kernel Matrix $M$ is symmetric positive semi-definite.
2.3.3 Variations of SVR

In this section we summarize the standard SVR problem and talk about its variations.

Recall equation (2.3.12), (2.3.23):

\[ w = \sum_{i=1}^{n} (d_i - d_i^*) x_i \]

\[ f(x) = wx + b = \sum_{i=1}^{n} (d_i - d_i^*) \langle x_i, x \rangle + b \]

If we consider the problem in the feature space \( \phi(X) \), not in the original input space \( X \). Then:

\[ w = \sum_{i=1}^{n} (d_i - d_i^*) \phi(x_i) \] (2.3.27)

\[ f(x) = w\phi(x) + b = \sum_{i=1}^{n} (d_i - d_i^*) \langle \phi(x_i), \phi(x) \rangle + b \]

\[ = \sum_{i=1}^{n} (d_i - d_i^*) \mathcal{K}(x_i, x) + b \] (2.3.28)

So we have a nonlinear function \( f \) and now the goal is to find the flattest function in the feature space, not in the original input space, that best approximates the data.

With the standard \( \varepsilon \)-insensitive loss function, the objective function is:

\[
\frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_{\varepsilon}
\]

\[
= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_i - d_i^*) (d_j - d_j^*) \langle \phi(x_i), \phi(x_j) \rangle + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_{\varepsilon}
\]

\[
= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_i - d_i^*) (d_j - d_j^*) \mathcal{K}(x_i, x_j) + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_{\varepsilon}
\] (2.3.29)

The standard SVR problem can be written as:

\[
\min_{b,d_i,d_i^*,\xi_i,\xi_i^*} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_i - d_i^*) (d_j - d_j^*) \mathcal{K}(x_i, x_j) + \lambda \sum_{i=1}^{n} (\xi_i + \xi_i^*)
\]

\[
\text{s.t.} \quad \begin{cases}
  y_i - \sum_{j=1}^{n} (d_j - d_j^*) \mathcal{K}(x_j, x_i) - b \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
  \sum_{j=1}^{n} (d_j - d_j^*) \mathcal{K}(x_j, x_i) + b - y_i \leq \varepsilon + \xi_i^*, & i = 1, \ldots, n. \\
  \xi_i, \xi_i^* \geq 0, & i = 1, \ldots, n.
\end{cases}
\] (2.3.30)
It is easy to see that we would need a Mercer kernel to have the first part of the objective function nonnegative.

The dual form of the standard SVR problem is:

$$\max_{d_i, d_i^*} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_i - d_i^*)(d_j - d_j^*)K(x_i, x_j)$$

$$- \sum_{i=1}^{n} (d_i + d_i^*)\varepsilon - \sum_{i=1}^{n} (d_i^* - d_i)y_i$$

$$\text{s.t.} \begin{cases} \sum_{i=1}^{n} (d_i^* - d_i) = 0, \ i = 1, \ldots, n. \\ 0 \leq d_i \leq \lambda, \ i = 1, \ldots, n. \\ 0 \leq d_i^* \leq \lambda, \ i = 1, \ldots, n. \end{cases}$$

There is another popular version of SVR called Linear Programming Support Vector Regression (LPSVR). Instead of choosing the flattest function which best fits the data, researchers propose to find $w$ that is contained in the smallest convex combination of the original input space $X$ or the feature input space $\phi(X)$ Smola and Schölkopf (2004).

Recall equation (2.3.27), (2.3.28):

$$w = \sum_{i=1}^{n} (d_i - d_i^*)\phi(x_i)$$

$$f = w\phi(x) + b = \sum_{i=1}^{n} (d_i - d_i^*)\langle \phi(x_i), \phi(x) \rangle + b$$

$$= \sum_{i=1}^{n} (d_i - d_i^*)K(x_i, x) + b$$

The objective function becomes Smola and Schölkopf (2004):

$$\sum_{i=1}^{n} |d_i - d_i^*| + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|\varepsilon$$
The LPSVR problem can be written as:

\[
\min_{b, d_i, d_i^*, \xi_i, \xi_i^*} \sum_{i=1}^{n} |d_i - d_i^*| + \lambda \sum_{i=1}^{n} (\xi_i + \xi_i^*) \\
\text{s.t.} \quad \begin{cases} 
    y_i - \sum_{j=1}^{n} (d_j - d_j^*) K(x_j, x_i) - b \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
    \sum_{j=1}^{n} (d_j - d_j^*) K(x_j, x_i) + b - y_i \leq \varepsilon + \xi_i^*, & i = 1, \ldots, n. \\
    \xi_i, \xi_i^* \geq 0, & i = 1, \ldots, n.
\end{cases}
\]  

(2.3.33)

Here we do not have the inner product term in the objective function and thus in this case researchers have proposed to use more general kernels which may not satisfy the Mercer Condition Burges (1998).

Recall equation (2.3.27), (2.3.28):

\[
w = \sum_{i=1}^{n} (d_i - d_i^*) \phi(x_i) \\
f(x) = w \phi(x) + b = \sum_{i=1}^{n} (d_i - d_i^*) \langle \phi(x_i), \phi(x) \rangle + b \\
= \sum_{i=1}^{n} (d_i - d_i^*) K(x_i, x) + b
\]

We use \((d_i - d_i^*)\) as the coefficients here because of the derivation of the dual formulation. From equation (2.3.6) we have: \(d_i \geq 0, d_i^* \geq 0\), but no restrictions on \((d_i - d_i^*)\) itself.

Later we are going to use a more general form:

\[
w = \sum_{i=1}^{n} \alpha_i \phi(x_i) 
\]

(2.3.34)

\[
f(x) = w \phi(x) + b = \sum_{i=1}^{n} \alpha_i \langle \phi(x_i), \phi(x) \rangle + b \\
= \sum_{i=1}^{n} \alpha_i K(x_i, x) + b
\]

(2.3.35)

to formulate the estimation problem.
2.4 Semi-infinite Programming and Cutting Plane Method

In this section we talk about the semi-infinite programming which always occurs when we incorporate some continuous constraints into the kernel based optimization problem. An algorithm called the Cutting Plane Method (CPM) is reviewed and later used in our proposed methods both theoretically and numerically Sun et al. (2010).

Semi-infinite programming is defined as in Hettich and Kortanek (1993):

\[
\min_{x \in X} \ f(x) \\
\text{s.t.} \ \begin{cases} 
  h(x) \leq 0 \\
  g(x, y) \leq 0, \ \forall y \in Y \end{cases} 
\]  

(2.4.1)

where \( f : R^n \rightarrow R \), \( h : R^n \rightarrow R \), \( g : R^n \times R^m \rightarrow R \), \( X \subseteq R^n \), \( Y \subseteq R^m \).

Notice that the constraint \( g(x, y) \) has a continuous variable \( y \) which does not appear in the objective function. This can be viewed as a special case of bilevel programs. And this constraint will result in infinite number of inequalities. A prior discretization strategy such as choosing some knots manually or generate some \( y \) randomly would reduce the constraints to finite number of inequalities and provide a way to solve the problem. But it cannot guarantee that the solution fully satisfied the continuous constraints, especially between the chosen knots.

Cutting Plane Method (CPM) which discretizes the continuous constraint and solves the optimization problem iteratively ensures that the constraint is strictly satisfied by the final solution Sun et al. (2010). It can be viewed as a posterior discretization method.

By introducing a positive variable \( \varepsilon \), the algorithm of CPM is:

- Step 1: Denote the constraints in the semi-infinite programming problem (2.4.1) without the continuous one as \( M_0 \), determine a tolerance \( \varepsilon > 0 \) and set \( k = 0 \).

- Step 2: Solve

\[
\min_{x \in X} \ f(x) \\
\text{s.t.} \ x \in M_k
\]
and denote the solution as $x^*_k$.

- Step 3: Solve

$$\max_{y \in Y} g(x^*_k, y)$$

and denote the solution as $y^*_k$ and the objective function value as $g(x^*_k, y^*_k)$.

- Step 4: If $g(x^*_k, y^*_k) \leq 0$, then stop and return the solution of the semi-infinite programming problem (2.4.1) as $x^*_k$. Otherwise set

$$M_{k+1} = M_k \cap \{x \in X : g(x, y^*_k) + \varepsilon \leq 0\}$$

- Step 5: Set $k + 1 = k$ and go to step 2.

This algorithm also reduce the continuous constraint to a finite number of inequalities and the theoretic result is as follows:

**Theorem 2** (Convergency of CPM Sun et al. (2010)). For any tolerance $\varepsilon > 0$, there exists an integer $N$, such that $g(x^*_N, y^*_N) \leq 0$ holds, where $\varepsilon$ and $g$ refer to the parameter and function in the algorithm of CPM.

By this convergency theorem we know that after a finite number of iterations, the semi-infinite programming problem (2.4.1) can be reduced to an optimization problem with finite number of constraints and a solution $x^*_k$ to the original problem can be obtained by solving the equivalent reduced optimization problem.
CHAPTER 3. ESTIMATION OF THE RND BY LPSVR

In this chapter we formulate the estimation of the RND into an optimization problem based on Support Vector Regression (SVR) using Linear Programming (LP). We propose two schemes in terms of different loss functions, the standard \( \varepsilon \)-insensitive loss function with a common tube size and the modified \( \varepsilon_i \)-insensitive loss function with a varying tube size. We prove that under the framework of linear programming we are guaranteed to obtain a global solution. Monte-Carlo simulations are conducted to check the performance of the proposed schemes. And we show that with a varying tube size we can better reduce the noises and obtain a solution with less bias. The SML method is also executed to serve as a benchmark.

The rest of the chapter is organized as follows. Section 3.1 discusses the formulation of the optimization problem with \( \varepsilon \)-insensitive loss function. Globalness of the solution is proved. Section 3.2 reviews the implementation steps of the benchmark method SML. Section 3.3 explains the designed Monte-Carlo simulation in details. Some standards of measurement are introduced in Section 3.4 and the performances of LPSVR and SML are presented. In section 3.5 we move one step further to formulate the optimization problem with \( \varepsilon_i \)-insensitive loss function which has a varying tube size. Global solution is also guaranteed under this scheme. Section 3.6 shows that with this varying tube size we can improve the accuracy level and in total \( \varepsilon_i \)-LPSVR has a better performance.

3.1 LPSVR

Now we have all the background knowledge ready to formulate the estimation of the RND into an optimization problem under the framework of Linear Programming (LP) based on SVR. We name the proposed scheme in this section as LPSVR.
Let \( \{(x_1, c_1), \ldots, (x_n, c_n)\} \) be the strike prices and the corresponding call option prices in the market, where \( x_i \geq 0, \ c_i \geq 0, \ i = 1 \ldots n \). The estimation problem is to find the RND \( f(x) \) that best approximates these data points and also as flat as possible.

Assume the RND:

\[
f(x) = \sum_{i=1}^{n} \alpha_i \mathcal{K}(x_i, x) + b \tag{3.1.1}
\]

where \( x \in [0, \infty) \).

To ensure the flatness of the RND in a linear form, the objective function is:

\[
\sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} L(y_i, f(x_i)) \tag{3.1.2}
\]

where \( L(y_i, f(x_i)) \) is the loss function describes how the function \( f(x) \) approximates these data points.

Here we choose the standard \( \varepsilon \)-insensitive loss function:

\[
|\xi_i|_{\varepsilon} := \begin{cases} 
0, & \text{if } |\xi_i| \leq \varepsilon \\
|\xi_i| - \varepsilon, & \text{otherwise}
\end{cases}
\]

The objective function becomes:

\[
\sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_{\varepsilon} \tag{3.1.3}
\]

where \( y_i \) denotes the real RND.

Notice that we do not have the real RND \( y_i \) directly. Instead we have the option prices \( c_i \) in the market. So we change the objective function by replacing \( (y_i - f(x_i)) \) with \( (c_i - C(x_i)) \). According to equation (2.1.1)

\[
C(K) = e^{-rt} \int_{K}^{\infty} (S - K)f(S)dS \tag{3.1.4}
\]

So the objective function becomes:

\[
\sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} |c_i - C(x_i)|_{\varepsilon} \tag{3.1.5}
\]
For a specified $\varepsilon$, incorporating no-arbitrage and the RND constraints from equation (2.1.15), we can formulate the estimation problem as:

$$\min_{b, \alpha, \xi, \xi^*} \sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} (\xi_i + \xi_i^*)$$

subject to

$$\begin{cases}
c_i - C(x_i) \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
C(x_i) - c_i \leq \varepsilon + \xi_i^*, & i = 1, \ldots, n. \\
\xi_i, \xi_i^* \geq 0, & i = 1, \ldots, n. \\
f(K) \geq 0, & K \in [0, \infty) \\
\int_{0}^{\infty} f(S)dS = 1 \\
C(0) = S_0 e^{-\delta t}
\end{cases}$$

(3.1.6)

Note that $c_i - C(x_i)$ is either non-positive or non-negative, so from the idea of $\varepsilon$-insensitive loss function ($\varepsilon \geq 0$), one of $\xi_i, \xi_i^*$ is going to be 0. We can just keep one of $\xi_i, \xi_i^*$ to have less variables. Besides introducing slack variables $d_i$, where $d_i \geq 0$, can help us rewrite the estimation problem without absolute values. So our problem becomes:

$$\min_{b, \alpha, d, \xi, \xi^*} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i$$

subject to

$$\begin{cases}
-\varepsilon - \xi_i \leq c_i - C(x_i) \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
-d_i \leq \alpha_i \leq d_i, & i = 1, \ldots, n. \\
f(K) \geq 0, & K \in [0, \infty) \\
\int_{0}^{\infty} f(S)dS = 1 \\
C(0) = S_0 e^{-\delta t}
\end{cases}$$

(3.1.7)

The first two constraints imply $\xi_i \geq 0, d_i \geq 0, i = 1, \ldots, n.$

Notice here we are using the variations of SVR, a linear objective function which does not involve the inner product. So in this case we can pick a more general kernel which may not satisfy the Mercer Condition and we choose to set $b = 0$ for simplicity. The kernel we pick should be supported
from $[0, \infty)$ and have nice integration properties. For these reasons we pick the log-logistic function as our kernel:

$$K(x_i, x) = \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2} \tag{3.1.8}$$

where $\beta > 0$ is the shape parameter.

We prove the following theorem for the optimization problem (3.1.7)

**Theorem 3.** For the log-logistic kernel: $K(x_i, x) = \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2}$, $x \in [0, \infty)$, and $b = 0$, the optimization problem (3.1.7) has a global solution.

**Proof.** If $K(x_i, x) = \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2}$, $x \in [0, \infty)$, $b = 0$, recall equation (3.1.1), (3.1.4):

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) + b$$

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K)f(S)dS$$

So:

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

$$= \sum_{i=1}^{n} \alpha_i \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2} \tag{3.1.9}$$

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K)f(S)dS$$

$$= e^{-rt} \int_{K}^{\infty} (S - K) \sum_{i=1}^{n} \alpha_i K(x_i, S)dS$$

$$= e^{-rt} \int_{K}^{\infty} (S - K) \sum_{i=1}^{n} \alpha_i \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1 + (S/x_i)^{\beta})^2} dS \tag{3.1.10}$$

$$= e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{K}^{\infty} (S - K) \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1 + (S/x_i)^{\beta})^2} dS$$

The log-logistic function is a probability density function. It integrates to one itself:

$$\int_{0}^{\infty} K(x_i, S)dS = \int_{0}^{\infty} \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1 + (S/x_i)^{\beta})^2} dS = 1, \ i = 1, \ldots, n.$$ 

and its mean is:

$$\int_{0}^{\infty} S K(x_i, S)dS = \int_{0}^{\infty} S \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1 + (S/x_i)^{\beta})^2} dS = \frac{x_i \pi/\beta}{\sin(\pi/\beta)}, \ i = 1, \ldots, n.$$
So:

\[
C(0) = e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{0}^{\infty} SK(x_i, S) dS = e^{-rt} \sum_{i=1}^{n} \alpha_i \frac{x_i \pi / \beta}{\sin(\pi / \beta)} = \frac{e^{-rt} \pi / \beta}{\sin(\pi / \beta)} \sum_{i=1}^{n} \alpha_i x_i
\]

The constraint on the RND \( f \) becomes:

\[
f(K) \geq 0 \iff \sum_{i=1}^{n} \alpha_i (\beta / x_i)(K / x_i)^{\beta - 1} \geq 0, \quad K \in [0, \infty)
\]

\[
\int_{0}^{\infty} f(S) dS = 1 \iff \int_{0}^{\infty} \sum_{i=1}^{n} \alpha_i K(x_i, S) dS = 1
\]

\[
\iff \sum_{i=1}^{n} \alpha_i \int_{0}^{\infty} \mathcal{K}(x_i, S) dS = 1 \quad (3.1.11)
\]

\[
\iff \sum_{i=1}^{n} \alpha_i = 0
\]

and the constraint on \( C(0) \) becomes:

\[
C(0) = S_0 e^{-\delta t} \iff \frac{e^{-rt} \pi / \beta}{\sin(\pi / \beta)} \sum_{i=1}^{n} \alpha_i x_i = S_0 e^{-\delta t} \quad (3.1.12)
\]

The optimization problem (3.1.7) becomes:

\[
\min_{\alpha_i, d_i, \xi_i} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i
\]

\[
\begin{align*}
-\varepsilon - \xi_i & \leq c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta / x_i)(S / x_i)^{\beta - 1}}{(1 + (S / x_i)^{\beta})^2} dS, \quad i = 1, \ldots, n. \\
c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta / x_i)(S / x_i)^{\beta - 1}}{(1 + (S / x_i)^{\beta})^2} dS & \leq \varepsilon + \xi_i, \quad i = 1, \ldots, n. \\
-d_i & \leq \alpha_i - d_i, \quad i = 1, \ldots, n. \\
\sum_{i=1}^{n} \alpha_i & \geq 0, \quad K \in [0, \infty) \\
\sum_{i=1}^{n} \alpha_i & = 1 \\
\frac{e^{-rt} \pi / \beta}{\sin(\pi / \beta)} \sum_{i=1}^{n} \alpha_i x_i & = S_0 e^{-\delta t}
\end{align*}
\]  

(3.1.13)

The variables of the estimation problem are \( \alpha_i, d_i, \xi_i, \quad i = 1, \ldots, n \). As we can see that the objective function is linear in terms of these variables. And the constrains are also linear in terms of these variables.
Notice in the third constraint we have a continuous variable $K$ which results in infinite inequalities. So we have formulated the estimation of the RND into a semi-infinite linear programming optimization problem.

Apply the cutting plane method (CPM) reviewed in section 2.4, the continuous constraint can be reduced to a finite number of inequalities. By theorem 2 the semi-infinite linear programming problem (3.1.13) can be reduced to an equivalent linear programming problem. Thus we are guaranteed to find a global solution.

$\lambda, \beta, \varepsilon$ are three positive parameters we need to determine before solving the optimization problem (3.1.13). $\lambda$ is the trade-off parameter between the flatness of the RND function and the goodness of fit of the data. $\beta$ is the shape parameter for the log-logistic kernel. $\varepsilon$ is the tube size used in the $\varepsilon$-insensitive loss function. We will use a data-driven method, i.e., the cross-validation method to determine these three parameters. Details are given in the later section.

### 3.2 SML

In section 2.2 we briefly introduced the SML method which we will use here as a benchmark.

We elaborate the implementation details as follows.

Notice that the data we have are the strike prices and their corresponding call option prices $\{(x_1, c_1), \ldots, (x_n, c_n)\}$ in the market, where $x_i \geq 0$, $c_i \geq 0$, $i = 1 \ldots n$.

The implementation steps of SML are:

- Convert the option prices to implied volatilities using Black-Scholes formula so we have $\{(x_1, \sigma_1), \ldots, (x_n, \sigma_n)\}$, where $x_i \geq 0$, $\sigma_i \geq 0$, $i = 1 \ldots n$.

- Fit the implied volatilities with the smoothing spline. The objective function is:

\[
\min \int \sum_{i=1}^{n} \left( \sigma_i - f(x_i) \right)^2 + \lambda \int f''(x)^2 \, dx
\]

where $f$ is the volatility function fitted by the natural cubic spline and $\lambda$ is a positive parameter which controls the trade-off between the smoothness of the function and the goodness of fit of the data. Cross-validation is used to find the best $\lambda$. 
• Convert the volatility function \( \hat{f} \) back to option prices \( C \) by Black-Scholes formula.

• Take the second derivative of the option price function with respect to the strike price and multiplying by the discounting factor, we obtain the RND as in equation (2.1.4):

\[
f(K) = e^{rt} \frac{\partial^2 C(K)}{\partial K^2}
\]

(3.2.2)

### 3.3 Monte-Carlo Simulations

In this section we discuss the designed Monte-Carlo simulations of solving the optimization problem (3.1.13) and evaluate the performance of our proposed method, LPSVR. The SML method is also executed to serve as a benchmark. We use the data of S&P 500 index options at the Chicago Board Options Exchange (CBOE) which are among the most actively traded financial derivatives in the world.

#### 3.3.1 Fit of the Real RND

Before jumping directly into sloving the optimization problem, we need to figure out how we evaluate the solution and moreover, how we evaluate the proposed method. To evaluate the solution, we would need the real RND so that we can compare it with the solution. To check the performance of our proposed method, we would solve the optimization problem multiple times and compare all the solutions with the real RND. Since we only have one data set, i.e., the strike prices and option prices, corresponding to one RND at one time. We would use the real RND to generate multiple option data sets.

Since the real RND is not observable directly, it is reasonable to fit the option prices in the market and take the fitted RND as the real one of the underlying asset to perform simulations. To allow more flexibilities, we use a combination of three log-normal density functions to construct the real RND.
Let \( \{ (x_1, c_1), \ldots, (x_n, c_n) \} \) be the strike prices and the corresponding S&P 500 index call option prices in the market, where \( x_i \geq 0, c_i \geq 0, i = 1 \ldots n \). Assume

\[
f(x) = \sum_{i=1}^{3} p_i \logn(x; \mu_i, \sigma_i)
\]

\[
= p_1 \logn(x; \mu_1, \sigma_1) + p_2 \logn(x; \mu_2, \sigma_2) + p_3 \logn(x; \mu_3, \sigma_3)
\]

where \( \logn(x; \mu, \sigma) \) is the probability density function of the log-normal distribution:

\[
\logn(x; \mu, \sigma) = \frac{1}{x \sigma \sqrt{2 \pi}} e^{-\frac{(\ln x - \mu)^2}{2 \sigma^2}}
\]

The objective function is:

\[
\sum_{i=1}^{n} (c_i - C(x_i))^2
\]

Incorporating no-arbitrage and the RND constraints from equation (2.1.15), we can formulate the fit problem as:

\[
\begin{align*}
\min_{p_i, \mu_i, \sigma_i} & \sum_{i=1}^{n} (c_i - C(x_i))^2 \\
\text{s.t.} & \ f(K) \geq 0, K \in [0, \infty) \\
& \int_{0}^{\infty} f(S) dS = 1 \\
& C(0) = S_0 e^{-\delta t}
\end{align*}
\]

The option price:

\[
C(K) = e^{-rt} \int_{K}^{\infty} (S - K) f(S) dS
\]

\[
= e^{-rt} \int_{K}^{\infty} (S - K) \sum_{i=1}^{3} p_i \logn(S; \mu_i, \sigma_i) dS
\]

\[
= e^{-rt} \sum_{i=1}^{3} p_i \int_{K}^{\infty} (S - K) \logn(S; \mu_i, \sigma_i) dS
\]
Plug in 0 for $K$ and evaluate the mean of log-logistic:

$$C(0) = e^{-rt} \int_0^\infty (S - 0) f(S) dS$$

$$= e^{-rt} \int_0^\infty S \sum_{i=1}^3 p_i \log n(S; \mu_i, \sigma_i) dS$$

$$= e^{-rt} \sum_{i=1}^3 p_i \int_0^\infty S \log n(S; \mu_i, \sigma_i) dS$$

$$= e^{-rt} \sum_{i=1}^3 p_i e^{\mu_i + \sigma_i^2}$$

where

$$\int_0^\infty S \log n(S; \mu, \sigma) dS = E(x|x \sim \log n(x; \mu, \sigma)) = e^{\mu + \sigma^2}$$

The constraint on $C(0)$ becomes:

$$C(0) = S_0 e^{-\delta t} \iff e^{-rt} \sum_{i=1}^3 p_i e^{\mu_i + \sigma_i^2} = S_0 e^{-\delta t}$$

Notice:

$$\log n(x; \mu, \sigma) = \frac{1}{x \sigma \sqrt{2\pi}} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}} \geq 0$$

$$\int_0^\infty \log n(S; \mu, \sigma) dS = 1 \quad (3.3.5)$$

We simply require all three parameter $p_1, p_2, p_3$ to be non-negative. So the constraints on the RND $f$ becomes:

$$f(K) \geq 0 \iff \sum_{i=1}^3 p_i \log n(K; \mu_i, \sigma_i) \geq 0 \iff p_1 \geq 0, \ p_2 \geq 0, \ p_3 \geq 0, \ K \in [0, \infty)$$

$$\int_0^\infty f(S) dS = 1 \iff \int_0^\infty \sum_{i=1}^3 p_i \log n(S; \mu_i, \sigma_i) dS = 1 \iff \sum_{i=1}^3 p_i = 1 \quad (3.3.6)$$
Introducing non-negative variables $\xi_i$, $i = 1, \ldots, n$, the fit problem becomes:

$\min_{p_i, \mu_i, \sigma_i, i = 1, 2, 3} \sum_{i=1}^{n} \xi_i^2$

s.t.

$-\xi_i \leq c_i - e^{-rt} \sum_{i=1}^{3} p_i \int_{x_i}^{\infty} (S - x_i) \log n(S; \mu_i, \sigma_i) dS \leq \xi_i, \quad i = 1, \ldots, n$

$p_i \geq 0, \quad i = 1, 2, 3$

$\sum_{i=1}^{3} p_i = 1$

$e^{-rt} \sum_{i=1}^{3} p_i e^{\mu_i + \sigma_i^2} = S_0 e^{-\delta t}$

This is a quadratic optimization problem with nonlinear constraints.

To better compare the performance of different methods, four cases of data with expiration days between one to two months, more than three months and more than six months are used. The market information for these four cases, i.e., the trading date, the maturity date, the time to maturity $t$, the current index price $S_0$, the risk free rate $r$ (3-month t-bill rate) and the divided yield rate $\delta$ are all listed in table 3.1.

<table>
<thead>
<tr>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trading Date</td>
<td>2017/06/05</td>
<td>2017/08/04</td>
<td>2017/06/05</td>
</tr>
<tr>
<td>Maturity Date</td>
<td>2017/07/28</td>
<td>2017/09/08</td>
<td>2017/10/31</td>
</tr>
<tr>
<td>Time to maturity $t$ (days)</td>
<td>53</td>
<td>35</td>
<td>148</td>
</tr>
<tr>
<td>Current index price $S_0$</td>
<td>2436.10</td>
<td>2476.83</td>
<td>2436.10</td>
</tr>
<tr>
<td>Risk free rate $r$ (%)</td>
<td>0.95</td>
<td>1.06</td>
<td>0.95</td>
</tr>
<tr>
<td>Dividend yield rate $\delta$ (%)</td>
<td>2.1</td>
<td>2.0</td>
<td>2.1</td>
</tr>
</tbody>
</table>

The fitted parameters for four cases are summarized in Table 3.2.

Figure 3.1, 3.2, 3.3, 3.4 show the fitting result graphically.

The fit is quiet good, therefore we can use the fitted RND to represent the real RND in the designed simulations.
Table 3.2: The fitted parameters of S&P 500 index option data by three log-normal density functions

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trading Date</strong></td>
<td>2017/06/05</td>
<td>2017/08/04</td>
<td>2017/06/05</td>
<td>2017/08/04</td>
</tr>
<tr>
<td><strong>Maturity Date</strong></td>
<td>2017/07/28</td>
<td>2017/09/08</td>
<td>2017/10/31</td>
<td>2018/03/16</td>
</tr>
<tr>
<td>$p_1$</td>
<td>0.0812</td>
<td>0.0823</td>
<td>0.0862</td>
<td>0.5934</td>
</tr>
<tr>
<td>$p_2$</td>
<td>0.0914</td>
<td>0.8115</td>
<td>0.3347</td>
<td>0.2894</td>
</tr>
<tr>
<td>$p_3$</td>
<td>0.8274</td>
<td>0.1062</td>
<td>0.5791</td>
<td>0.1172</td>
</tr>
<tr>
<td>$\mu_1$</td>
<td>7.8020</td>
<td>7.7669</td>
<td>7.5628</td>
<td>7.8594</td>
</tr>
<tr>
<td>$\mu_2$</td>
<td>7.7023</td>
<td>7.8176</td>
<td>7.7690</td>
<td>7.7779</td>
</tr>
<tr>
<td>$\mu_3$</td>
<td>7.8052</td>
<td>7.8165</td>
<td>7.8340</td>
<td>7.5688</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.0285</td>
<td>0.0543</td>
<td>0.2032</td>
<td>0.0369</td>
</tr>
<tr>
<td>$\sigma_2$</td>
<td>0.0987</td>
<td>0.0214</td>
<td>0.0599</td>
<td>0.0696</td>
</tr>
<tr>
<td>$\sigma_3$</td>
<td>0.0245</td>
<td>0.0247</td>
<td>0.0323</td>
<td>0.2090</td>
</tr>
</tbody>
</table>

3.3.2 Generation of Option Data Sets

Now the real RND is in hand, we can generate multiple option data sets for our optimization problem. First we generate the theoretical call option prices using:

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K)f(S)dS$$

The strike prices are set from 1500 to 3000 with a step size 25, in total 61 numbers.

In the real world, the option prices are going to deviate from the theoretical prices due to imperfect market conditions, such as non-synchronous trading, bid-ask spreads, market frictions and so on. To mimic the real world, we introduce random noises and add them to the theoretical option prices. To generate the random noises, it is reasonable to take the bid-ask spreads and the liquidity of options at different strike prices into account. The random noises $R_i$ are assumed to be uniformly distributed between negative and positive half of the spread multiplying by the liquidity factor, i.e.,

$$R_i \sim U\left(-\frac{S_i \times L_i}{2}, \frac{S_i \times L_i}{2}\right)$$

where $U$ is the uniform distribution, $S_i$ is the bid-ask spread at strike $x_i$, $L_i$ is the liquidity of the option at strike $x_i$. 
The bid-ask spread $S_i$ is assumed to be 5% of the minimum of the call option price $c_i$ and put option price $p_i$ at strike price $x_i$ with a floor 50 cents and a cap 5 dollars, i.e.,:

$$S_i = \begin{cases} 
5 & \text{if } S_i > 5 \\
5\% \times \min(c_i, p_i) & \text{if } 0.5 \leq S_i \leq 5 \\
0.5 & \text{if } S_i < 0.5 
\end{cases} \quad (3.3.8)$$

The liquidity factor $L_i$ is specified as in A"ıt-Sahalia and Duarte (2003):

$$L_i = 1 + 10 \times |x_i/S_0 - 1|$$

where $x_i$ is the strike price and $S_0$ is the current stock price.

Add the generated random noises to the theoretical option prices, we obtain one data set of strike prices and their corresponding option prices. Repeat the noise generation process for 1000 times, we would obtain 1000 data sets of strike prices and their corresponding option prices. For each case we generate 1000 data sets and solve the optimization problem to obtain 1000 estimators to the real RND.

3.3.3 Grid Search and Cross Validation of Extra Parameters

We have three extra parameters to determine before solving the optimization problem (3.1.13), the trade-off parameter between the flatness of the RND function and the goodness of fit of the data $\lambda$, the shape parameter for the log-logistic kernel $\beta$, the tube size used in the $\varepsilon$-insensitive loss function $\varepsilon$.

We start with a grid search and a data-driven method, i.e, the 10-fold cross-validation method is used to determine these three parameters. The whole process of solving the estimation problem for one of the cases is as follows:

- Step 1: Divide 61 data points into 10 groups with the first group containing 7 numbers and the rest containing 6 numbers. Within each group the strike prices are equally spaced with the step size 250.
Step 2: Determine the range of these three parameters for grid search. We use $2^i$, $i = 1, \ldots, 10$ for $\lambda$, 10 to 50 with step size 1 for $\beta$, 0 to $\max(s_i)$, $i = 1, \ldots, 53$ with step size 0.01 for $\varepsilon$ where $s_i$ is the option’s bid-ask spread at strike price $x_i$.

Step 3: Start with a set of value of three parameters and use 9 groups of data to solve the optimization problem (3.1.13). Evaluate the absolute deviation of the option prices on the rest group.

Step 4: Take turns to leave a group of data out to solve the optimization problem (3.1.13) and evaluate the absolute deviation of option prices on that reserved group using the same set of parameters as Step 3. In total we solved the optimization problem 10 times.

Step 5: Calculate the mean of the absolute deviation of option prices for the set of parameters.

Step 6: Start with another set of value of three parameters and repeat from Step 3 to Step 6 until the grid search is finished.

Step 7: Pick the set of parameters with the minimum mean absolute deviation of the option prices and resolve the optimization problem (3.1.13) with 10 groups of data to obtain the estimated RND.

Step 8: Repeat Step 1 to Step 7 with 1000 different generated data sets and obtain 1000 estimated RNDs.

We repeat the whole process for four cases and after that, we have obtained 1000 estimated RNDs for the real RND in each case.

### 3.4 Measurements and Results

To compare the performance of the proposed method with the SML method, the accuracy and the stability of the estimated RND will be measured. In the context of the density estimation, a measure called Root Integrated Mean Squared Error (RIMSE) will be used.
It is well known that Mean Squared Error (MSE) of an estimator can be separated into the Squared Bias (SB) and the Variance (V) of the estimator, i.e., \( \text{MSE} = \text{SB} + \text{V} \). As used in the density estimation valuation, these measures are integrated and taken the square root. So Root Integrated Mean Squared Error (RIMSE) can also be separated into Root Integrated Squared Bias (RISB) and Root Integrated Variance (RIV), i.e., \( \text{RIMSE}^2 = \text{RISB}^2 + \text{RIV}^2 \), such that:

\[
\text{RIMSE} = \left( \int_{0}^{\infty} E[(\hat{f}(x) - f(x))^2] \, dx \right)^{1/2}
\]

\[
\text{RISB} = \left( \int_{0}^{\infty} [(E\hat{f}(x) - f(x))^2] \, dx \right)^{1/2}
\]

\[
\text{RIV} = \left( \int_{0}^{\infty} E[(\hat{f}(x) - E\hat{f}(x))^2] \, dx \right)^{1/2}
\]

where \( \hat{f}(x) \) is the estimated RND and \( f(x) \) is the real RND.

Similar to MSE, SB and V, RIMSE is also a measure of the overall quality of the estimator, RISB is a measure of the accuracy and RIV is a measure of the stability.

The fitting results of the optimization problem (3.1.13) are shown in Figure 3.5, 3.6, 3.7, 3.8, where we compare the performance between SML and LPSVR by showing the fitted RND graphically.

The measurement results are shown in Table 3.3.

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SML</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{RIMSE} )</td>
<td>0.2265</td>
<td>0.2221</td>
<td>0.2008</td>
<td>0.2132</td>
</tr>
<tr>
<td>( \text{RISB} )</td>
<td>0.0787</td>
<td>0.1631</td>
<td>0.0965</td>
<td>0.1317</td>
</tr>
<tr>
<td>( \text{RIV} )</td>
<td>0.2124</td>
<td>0.1507</td>
<td>0.1761</td>
<td>0.1677</td>
</tr>
<tr>
<td><strong>LPSVR</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{RIMSE} )</td>
<td>0.1579</td>
<td>0.1428</td>
<td>0.1378</td>
<td>0.1540</td>
</tr>
<tr>
<td>( \text{RISB} )</td>
<td>0.0425</td>
<td>0.0747</td>
<td>0.0563</td>
<td>0.0706</td>
</tr>
<tr>
<td>( \text{RIV} )</td>
<td>0.1521</td>
<td>0.1217</td>
<td>0.1258</td>
<td>0.1369</td>
</tr>
</tbody>
</table>

To better interpret Table 3.3, Figure 3.9, 3.10, 3.11, 3.12 are generated. Compared with SML, LPSVR has a smaller RIMSE. The RISB and RIV of the LPSVR are also smaller than SML. In total LPSVR outperforms SML in both accuracy and stability.
3.5 \( \varepsilon_i \)-LPSVR

We name the proposed scheme in this section as \( \varepsilon_i \)-LPSVR.

In section 3.1 we formulate the estimation of the RND into a semi-infinite linear programming problem with the \( \varepsilon \)-insensitive loss function.

\[
|\xi_i|_{\varepsilon} := \begin{cases} 
0, & \text{if } |\xi_i| \leq \varepsilon \\
|\xi_i| - \varepsilon, & \text{otherwise}
\end{cases}
\]

The idea of this loss function is to set a common tube size, penalize the points outside the tube and neglect points inside the tube. The penalty is counted in a linear form, i.e., the distance from the outside point to the closest boundary of the tube.

The advantage of the loss function is the sparsity brought by the tube. The tube size is determined by grid search and cross validation in section 3.3. Other efforts to find the optimal tube size \( \varepsilon \) in the literature includes the \( v \)-SVR which incorporates the tube size \( \varepsilon \) as a variable in the objective function Schölkopf et al. (1999). And the authors in Schölkopf et al. (1999) raised a parametric model where we can use an arbitrary shape instead of a tube. Although the SVR problem is written in terms of an arbitrary shape loss function, however, it is hardly implemented in the optimization problem. Either the noise levels at different data points are uniform, i.e., meaning that we cannot distinguish them or we know the noise levels are different but cannot easily determine them. To determine a common threshold \( \varepsilon \) has already involved a certain amount of calculation, let alone different penalty thresholds for different data points.

The situation for the estimation of the RND is different. The data we have are the strike prices and the corresponding option prices. It would be reasonable to have different penalty thresholds at different strike points because one should not assume a constant noise level across the data. And we know that the market noises are closely related to the bid-ask spreads. Inspired by the standard \( \varepsilon \)-insensitive loss function, we define the following \( \varepsilon_i \)-insensitive loss function:

\[
|\xi_i|_{\varepsilon_i} := \begin{cases} 
0, & \text{if } |\xi_i| \leq \varepsilon_i \\
|\xi_i| - \varepsilon_i, & \text{otherwise}
\end{cases}
\]
\varepsilon_i := vS_i

where $S_i$ is the bid-ask spread at strike price $x_i$ that we have in the market and we restrict the proportion parameter $v \in (0, 1/2)$. If $v = 0$, then we penalize each data point by the $L_1$ norm. If $v = 1/2$, then we set the threshold for each data point as half of its bid-ask spread, meaning that we allow it to deviate up or down from the observed option price by half of its spread.

By setting $\varepsilon_i := vS_i$, the penalty threshold is closely related to the noise level. By searching for the proportion parameter $v$, instead of a common tube size $\varepsilon$, we gain more flexibility within the same amount of computation. Besides, we obtain variable penalty thresholds while avoiding exhaustively searching for arbitrary shapes.

The objective function is:

$$\sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} |c_i - C(x_i)|\varepsilon_i$$  \hspace{1cm} (3.5.1)

For a specified $v$, i.e., specified $\varepsilon_i$, $i = 1, 2, \ldots, n$, incorporating no-arbitrage and RND constraints from equation (2.1.15), we can formulate the estimation problem as:

$$\min_{b, \alpha_i, \xi_i, \xi^*_i} \sum_{i=1}^{n} |\alpha_i| + \lambda \sum_{i=1}^{n} (\xi_i + \xi^*_i)$$

s.t. \hspace{1cm}

$$c_i - C(x_i) \leq \varepsilon_i + \xi_i, \quad i = 1, \ldots, n.$$  

$$C(x_i) - c_i \leq \varepsilon_i + \xi^*_i, \quad i = 1, \ldots, n.$$  

$$\xi_i, \xi^*_i \geq 0, \quad i = 1, \ldots, n.$$  

$$f(K) \geq 0, \quad K \in [0, \infty)$$  

$$\int_0^\infty f(S)dS = 1$$  

$$C(0) = S_0e^{-\delta t}$$  \hspace{1cm} (3.5.2)
As in section 3.1, the problem can be rewritten as:

\[
\min_{b, \alpha_i, d_i, \xi_i} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i
\]

s.t.

\[
\begin{align*}
-\varepsilon_i - \xi_i &\leq c_i - C(x_i) \leq \varepsilon_i + \xi_i, &i = 1, \ldots, n. \\
-d_i &\leq \alpha_i \leq d_i, &i = 1, \ldots, n. \\
f(K) &\geq 0, &K \in [0, \infty) \\
\int_{0}^{\infty} f(S)dS &= 1 \\
C(0) &= S_0e^{-\delta t}
\end{align*}
\]

(3.5.3)

The first two constraints imply \(\xi_i \geq 0, d_i \geq 0, i = 1, \ldots, n\).

We set \(b = 0\) for simplicity and choose the log-logistic function as our kernel:

\[
\mathcal{K}(x_i, x) = \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2}
\]

(3.5.4)

where \(\beta > 0\) is the shape parameter.

By a very similar argument as in theorem 3, we can prove the following theorem for the optimization problem (3.5.3).

**Theorem 4.** For the log-logistic kernel: \(\mathcal{K}(x_i, x) = \frac{(\beta/x_i)(x/x_i)^{\beta-1}}{(1 + (x/x_i)^{\beta})^2}, x \in [0, \infty), \) and \(b = 0\), the optimization problem (3.5.3) has a global solution.

**Proof.** Replace \(\varepsilon\) by \(\varepsilon_i\) in the optimization problem (3.1.13) and by the same argument as in theorem 3, we can prove theorem 4. \(\square\)
The optimization problem (3.5.3) finally becomes:

$$\min_{\alpha_i, d_i, \xi_i} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i$$

subject to

$$-\varepsilon_i - \xi_i \leq c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S-x_i) \left(\frac{\beta}{x_j} \right) \left(\frac{S}{x_j}\right)^{\beta-1} \frac{dS}{1+(S/x_j)^{\beta}} \quad i = 1, \ldots, n.$$  

$$c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S-x_i) \left(\frac{\beta}{x_j} \right) \left(\frac{S}{x_j}\right)^{\beta-1} \frac{dS}{1+(S/x_j)^{\beta}} \leq \varepsilon_i + \xi_i, \quad i = 1, \ldots, n.$$  

$$-d_i \leq \alpha_i \leq d_i, \quad i = 1, \ldots, n.$$  

$$\sum_{i=1}^{n} \alpha_i \left(\frac{\beta}{x_i} \right) \frac{(K/x_i)^{\beta-1}}{1+(K/x_i)^{\beta}} \geq 0, \quad K \in [0, \infty)$$  

$$\sum_{i=1}^{n} \alpha_i = 1$$  

$$\frac{e^{-rt} \pi^2}{\sin(\pi/\beta)} \sum_{i=1}^{n} \alpha_i x_i = S_0 e^{-\delta t}$$

The variables of the estimation problem are $\alpha_i, d_i, \xi_i, i = 1, \ldots, n$. As we can see that the objective function is linear in terms of these variables. And the constrains are also linear in terms of these variables. In the third constraint we have a continuous variable $K$ which results in infinite inequalities. So it is also a semi-infinite linear programming problem.

$\lambda, \beta, v$ are three positive parameters we need to determine before solving the optimization problem (3.5.5). $\lambda$ is the trade-off parameter between the flatness of the RND function and the goodness of fit of the data. $\beta$ is the shape parameter for the log-logistic kernel. $v$ is the proportion parameter used in the $\varepsilon_i$-insensitive loss function, where $\varepsilon_i := vS_i$. We will use a data-driven method, i.e, the cross-validation method to determine these three parameters. Details are given in the later section.

### 3.6 Simulation Results

The simulation process for the optimization problem (3.5.5) is the same for the optimization problem (3.1.13). We are using the same generated data here so we can compare the performance among SML, LPSVR and $\varepsilon_i$-LPSVR.

There is one parameter that needs to be addressed separately, i.e., the proportion parameter $v$. We just replace the search for $\varepsilon$ by the search for $v$ using $\frac{1}{n}$, where $n$ is a integer and $n = 2, \ldots, 100$. 
The fitting results of the optimization problem (3.5.5) are shown in Figure 3.13, 3.14, 3.15, 3.16, where we compare the performance between LPSVR and ε-LPSVR by showing the fitted RND graphically.

The measurement results are shown in Table 3.4.

Table 3.4: The RIMSEs, RISBs and RIVs of SML, LPSVR and ε-LPSVR

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SML</td>
<td>RIMSE</td>
<td>0.2265</td>
<td>0.2221</td>
<td>0.2008</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0787</td>
<td>0.1631</td>
<td>0.0965</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.2124</td>
<td>0.1507</td>
<td>0.1761</td>
</tr>
<tr>
<td>LPSVR</td>
<td>RIMSE</td>
<td>0.1579</td>
<td>0.1428</td>
<td>0.1378</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0425</td>
<td>0.0747</td>
<td>0.0563</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.1521</td>
<td>0.1217</td>
<td>0.1258</td>
</tr>
<tr>
<td>ε-LPSVR</td>
<td>RIMSE</td>
<td>0.1396</td>
<td>0.1283</td>
<td>0.1178</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0303</td>
<td>0.0386</td>
<td>0.0364</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.1363</td>
<td>0.1224</td>
<td>0.1120</td>
</tr>
</tbody>
</table>

To better interpret Table 3.4, Figure 3.17, 3.18, 3.19, 3.20 are generated. Compared with SML, ε-LPSVR outperforms both in the accuracy and stability level. Compared with LPSVR, ε-LPSVR improves apparently in RISB and maintains the level in RIV and thus in total a smaller RIMSE. Generally ε-LPSVR outperforms LPSVR with the same stability level and an improvement in the accuracy level.
Figure 3.1: The fitting result of S&P 500 index option data by three log-normal density functions for Case 1.

Figure 3.2: The fitting result of S&P 500 index option data by three log-normal density functions for Case 2.
Figure 3.3: The fitting result of S&P 500 index option data by three log-normal density functions for Case 3.

Figure 3.4: The fitting result of S&P 500 index option data by three log-normal density functions for Case 4.
Figure 3.5: Estimated RND of S&P 500 index option data by SML (left) and LPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.

Figure 3.6: Estimated RND of S&P 500 index option data by SML (left) and LPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.
Figure 3.7: Estimated RND of S&P 500 index option data by SML (left) and LPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.

Figure 3.8: Estimated RND of S&P 500 index option data by SML (left) and LPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by SML or LPSVR.
Figure 3.9: The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 1.

Figure 3.10: The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 2.
Figure 3.11: The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 3.

Figure 3.12: The RIMSEs, RISBs and RIVs of estimated RNDs by SML and LPSVR for Case 4.
Figure 3.13: Estimated RND of S&P 500 index option data by LPSVR (left) and $\varepsilon_i$-LPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by LPSVR or $\varepsilon_i$-LPSVR.

Figure 3.14: Estimated RND of S&P 500 index option data by LPSVR (left) and $\varepsilon_i$-LPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by LPSVR or $\varepsilon_i$-LPSVR.
Figure 3.15: Estimated RND of S&P 500 index option data by LPSVR (left) and $\varepsilon$-LPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by LPSVR or $\varepsilon$-LPSVR.

Figure 3.16: Estimated RND of S&P 500 index option data by LPSVR (left) and $\varepsilon$-LPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by LPSVR or $\varepsilon$-LPSVR.
Figure 3.17: The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 1.

Figure 3.18: The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 2.
Figure 3.19: The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 3.

Figure 3.20: The RIMSEs, RISBs and RIVs of estimated RNDs by SML, LPSVR and $\varepsilon_i$-LPSVR for Case 4.
CHAPTER 4. ESTIMATION OF THE RND BY QPSVR

In this chapter we formulate the estimation of the RND into an optimization problem based on Support Vector Regression (SVR) using Quadratic Programming (QP). We propose two schemes in terms of different loss functions, $\varepsilon$-insensitive square loss function with a common tube size and $\varepsilon_i$-insensitive square loss function with a varying tube size. We prove that under the framework of quadratic programming we are guaranteed to obtain a unique solution. Monte-Carlo simulations are conducted to check the performance of the proposed schemes. And we show that with a varying tube size we can better reduce the noises and obtain a solution with less bias. The performance of the methods under the framework of LPSVR from chapter 3 are presented here to serve as a benchmark.

The rest of the chapter is organized as follows. Section 4.1 discusses the formulation of the optimization problem with $\varepsilon$-insensitive square loss function. Globalness and Uniqueness of the solution are proved. Section 4.2 reviews the methods proposed in chapter 3 and take them as benchmarks. Section 4.3 explains the designed Monte-Carlo simulation and Section 4.4 presents the simulation results of the performances among LPSVR, $\varepsilon_i$-LPSVR and QPSVR. In section 4.5 we move one step further to formulate the optimization problem with $\varepsilon_i$-insensitive square loss function which has a varying tube size. Global and unique solution is also guaranteed under this scheme. Section 4.6 shows that with this varying tube size we can improve the accuracy level and in total $\varepsilon_i$-QPSVR has the best performance among all four proposed methods.

4.1 QPSVR

Now we have all the background knowledge ready to formulate the estimation of the RND into an optimization problem under the framework of Quadratic Programming (QP) based on SVR. We name the proposed scheme in this section as QPSVR.
Let \( \{(x_1, c_1), \ldots, (x_n, c_n)\} \) be the strike prices and the corresponding call option prices in the market, where \( x_i \geq 0, \ c_i \geq 0, \ i = 1 \ldots n \). The estimation problem is to find the RND \( f(x) \) that best approximates these data points and also as flat as possible.

Assume the RND:

\[
    w = \sum_{i=1}^{n} \alpha_i \phi(x_i) \tag{4.1.1}
\]

\[
    f(x) = w \phi(x) + b = \sum_{i=1}^{n} \alpha_i \langle \phi(x_i), \phi(x) \rangle + b \tag{4.1.2}
\]

where \( x \in [0, \infty) \).

To ensure the flatness of the RND in a quadratic form, the objective function is:

\[
    \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} L(y_i, f(x_i)) \tag{4.1.3}
\]

where \( L(y_i, f(x_i)) \) is the loss function describes how the function \( f(x) \) approximates these data points.

In chapter 3 we choose the standard \( \varepsilon \)-insensitive loss function:

\[
    |\xi_i|_\varepsilon := \begin{cases} 
        0, & \text{if } |\xi_i| \leq \varepsilon \\
        |\xi_i| - \varepsilon, & \text{otherwise}
    \end{cases}
\]

Here we modify it to keep the advantage of sparsity and call it \( \varepsilon \)-insensitive square loss function:

\[
    |\xi_i|_\varepsilon^2 := \begin{cases} 
        0, & \text{if } |\xi_i| \leq \varepsilon \\
        (|\xi_i| - \varepsilon)^2, & \text{otherwise}
    \end{cases}
\]

The objective function becomes:

\[
    \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} |y_i - f(x_i)|_\varepsilon^2 \tag{4.1.4}
\]

where \( y_i \) denotes the real RND.
Notice that we do not have the real RND $y_i$ directly. Instead we have the option prices $c_i$ in the market. So we change the objective function by replacing $(y_i - f(x_i))$ with $(c_i - C(x_i))$, where:

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K)f(S) dS$$  \hspace{1cm} (4.1.5)

So the objective function becomes:

$$\frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} |c_i - C(x_i)|^2$$  \hspace{1cm} (4.1.6)

For a specified $\varepsilon$, incorporating no-arbitrage and the RND constraints from equation (2.1.15), we can formulate the estimation problem as:

$$\min_{b, \alpha, \xi, \xi^*} \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} (\xi_i^2 + (\xi_i^*)^2)$$

s.t. \hspace{1cm}

\begin{align*}
&c_i - C(x_i) \leq \varepsilon + \xi_i, \quad i = 1, \ldots, n. \\
&C(x_i) - c_i \leq \varepsilon + \xi_i^*, \quad i = 1, \ldots, n. \\
&\xi_i, \xi_i^* \geq 0, \quad i = 1, \ldots, n. \\
&f(K) \geq 0, \quad K \in [0, \infty) \\
&\int_{0}^{\infty} f(S) dS = 1 \\
&C(0) = S_0 e^{-\delta t}
\end{align*}  \hspace{1cm} (4.1.7)

Note that $c_i - C(x_i)$ is either non-positive or non-negative, so from the idea of $\varepsilon$-insensitive square loss function ($\varepsilon \geq 0$), one of $\xi_i, \xi_i^*$ is going to be 0. We can just keep one of $\xi_i, \xi_i^*$ to have less variables. Besides $w = \sum_{i=1}^{n} \alpha_i \phi(x_i)$. So our problem becomes:

$$\min_{b, \alpha, \xi, \xi^*} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) + \lambda \sum_{i=1}^{n} \xi_i^2$$

s.t. \hspace{1cm}

\begin{align*}
&-\varepsilon - \xi_i \leq c_i - C(x_i) \leq \varepsilon + \xi_i, \quad i = 1, \ldots, n. \\
&f(K) \geq 0, \quad K \in [0, \infty) \\
&\int_{0}^{\infty} f(S) dS = 1 \\
&C(0) = S_0 e^{-\delta t}
\end{align*}  \hspace{1cm} (4.1.8)
The first constraint implies $\xi_i \geq 0$, $i = 1, \ldots, n$.

Notice here we are using a quadratic objective function which involves the inner product. So in this case we should pick a kernel which needs to satisfy the Mercer Condition and we choose to set $b = 0$ for simplicity. The kernel we pick should be supported from $[0, \infty)$ and have nice integration properties. For these reasons we pick the Radial Basis Function kernel (RBF), also named as Gaussian kernel:

$$K(x_i, x) = \exp(-\gamma \|x - x_i\|^2)$$

where $\gamma > 0$ is the scale parameter.

We prove the following theorem for the optimization problem (4.1.8)

**Theorem 5.** For the RBF kernel: $K(x_i, x) = \exp(-\gamma \|x - x_i\|^2)$, $x \in [0, \infty)$, and $b = 0$ the optimization problem (4.1.8) has a global solution.

**Proof.** If $K(x_i, x) = \exp(-\gamma \|x - x_i\|^2)$, $x \in [0, \infty)$, $b = 0$, recall equation (4.1.2), (4.1.5):

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) + b$$

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K) f(S) dS$$

So:

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

$$= \sum_{i=1}^{n} \alpha_i e^{-\gamma(x-x_i)^2}$$

$$C(K) = e^{-rt} \int_{K}^{\infty} (S - K) f(S) dS$$

$$= e^{-rt} \int_{K}^{\infty} (S - K) \sum_{i=1}^{n} \alpha_i K(x_i, S) dS$$

$$= e^{-rt} \int_{K}^{\infty} (S - K) \sum_{i=1}^{n} \alpha_i e^{-\gamma(S-x_i)^2} dS$$

$$= e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{K}^{\infty} (S - K) e^{-\gamma(S-x_i)^2} dS$$

(4.1.10)
For the integration of the RBF kernel, we would need the Gauss Error Function which is defined as:

$$\text{erf}(x) := \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt, \quad x \geq 0$$

where \( \text{erf}(0) = 0, \text{erf}(\infty) = 1 \). Its integration is:

$$\int_0^\infty \mathcal{K}(x_i, S) dS = \int_0^\infty e^{-\gamma(S-x_i)^2} dS$$

$$\quad = \int_{-x_i}^\infty e^{-\gamma t^2} dt$$

$$\quad = \int_{-x_i}^0 e^{-\gamma t^2} dt + \int_0^\infty e^{-\gamma t^2} dt$$

$$\quad = \int_0^{x_i} e^{-\gamma t^2} dt + \frac{1}{\sqrt{\gamma}} \int_0^\infty e^{-u^2} du$$

$$\quad = \frac{1}{\sqrt{\gamma}} \int_0^{x_i \sqrt{\gamma}} e^{-u^2} du + \frac{1}{\sqrt{\gamma} \sqrt{\pi}}$$

$$\quad = \frac{1}{\sqrt{\gamma} \sqrt{\pi}} \text{erf}(x_i \sqrt{\gamma}) + \frac{1}{\sqrt{\gamma} \sqrt{\pi}}$$

$$\quad = \frac{1}{\sqrt{\gamma} \sqrt{\pi}} (\text{erf}(x_i \sqrt{\gamma}) + 1) \quad (4.1.12)$$

And its mean is:

$$\int_0^\infty S \mathcal{K}(x_i, S) dS = \int_0^\infty S e^{-\gamma(S-x_i)^2} dS$$

$$\quad = \int_{-x_i}^\infty (t + x_i) e^{-\gamma t^2} dt$$

$$\quad = \int_{-x_i}^\infty t e^{-\gamma t^2} dt + \int_{-x_i}^\infty x_i e^{-\gamma t^2} dt$$

$$\quad = \frac{1}{2\gamma} e^{-\gamma t^2} \bigg|_{t=-x_i}^{t=x_i} + x_i \int_{-x_i}^{x_i} e^{-\gamma t^2} dt$$

$$\quad = \frac{1}{2\gamma} e^{-\gamma x_i^2} + \frac{x_i \sqrt{\pi}}{\sqrt{\gamma} \sqrt{\pi}} (\text{erf}(x_i \sqrt{\gamma}) + 1)$$

$$\quad = \frac{1}{2\gamma} e^{-\gamma x_i^2} + \frac{x_i \sqrt{\pi}}{\sqrt{\gamma} \sqrt{\pi}} (\text{erf}(x_i \sqrt{\gamma}) + 1)$$

So:

$$C(0) = e^{-rt} \sum_{i=1}^n \alpha_i \int_0^\infty S \mathcal{K}(x_i, S) dS = e^{-rt} \sum_{i=1}^n \alpha_i \left( \frac{1}{2\gamma} e^{-\gamma x_i^2} + \frac{x_i \sqrt{\pi}}{\sqrt{\gamma} \sqrt{\pi}} (\text{erf}(x_i \sqrt{\gamma}) + 1) \right)$$

The constraint on the RND \( f \) becomes:

$$f(K) \geq 0 \iff \sum_{i=1}^n \alpha_i e^{-\gamma(K-x_i)^2} \geq 0, \quad K \in [0, \infty)$$
\[
\int_0^\infty f(S)dS = 1 \iff \int_0^\infty \sum_{i=1}^n \alpha_i \mathcal{K}(x_i, S)dS = 1
\]
\[
\iff \sum_{i=1}^n \alpha_i \int_0^\infty \mathcal{K}(x_i, S)dS = 1
\]
\[
\iff \sum_{i=1}^n \alpha_i \left( \frac{1}{\sqrt{\gamma \pi}} \left( \text{erf}(x_i \sqrt{\gamma}) + 1 \right) \right) = 1
\]

(4.1.14)

and the constraint on \(C(0)\) becomes:
\[
C(0) = S_0 e^{-\delta t} \iff e^{-rt} \sum_{i=1}^n \alpha_i \left( \frac{1}{2\gamma} e^{-\gamma x_i^2} + \frac{\sqrt{\pi}}{\sqrt{\gamma}} (\text{erf}(x_i \sqrt{\gamma}) + 1) \right) = S_0 e^{-\delta t}
\]

(4.1.15)

The optimization problem becomes:
\[
\min_{\alpha_i, \xi_i} \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j e^{-\gamma (x_i - x_j)^2} + \lambda \sum_{i=1}^n \xi_i^2
\]
\[
\begin{align*}
-\varepsilon - \xi_i & \leq c_i - e^{-rt} \sum_{i=1}^n \alpha_i \int_{x_i}^{\infty} (S - x_i) e^{-\gamma (S-x_i)^2} dS, & i = 1, \ldots, n. \\
\sum_{i=1}^n \alpha_i e^{-\gamma (x-x_i)^2} & \geq 0, & K \in [0, \infty) \\
\sum_{i=1}^n \alpha_i \left( \frac{1}{\sqrt{\gamma \pi}} \left( \text{erf}(x_i \sqrt{\gamma}) + 1 \right) \right) & = 1 \\
e^{-rt} \sum_{i=1}^n \alpha_i \left( \frac{1}{2\gamma} e^{-\gamma x_i^2} + \frac{\sqrt{\pi}}{\sqrt{\gamma}} (\text{erf}(x_i \sqrt{\gamma}) + 1) \right) & = S_0 e^{-\delta t}
\end{align*}
\]

(4.1.16)

The variables of the estimation problem are \(\alpha_i, \xi_i, i = 1, \ldots, n\). As we can see that the objective function is quadratic in terms of these variables. And the constrains are linear in terms of these variables.

Notice in the third constraint we have a continuous variable \(K\) which results in infinite inequalities. So we have formulated the estimation of the RND into a semi-infinite quadratic programming optimization problem.

Apply the cutting plane method (CPM) reviewed in section 2.4, the continuous constraint can be reduced to a finite number of inequalities. By theorem 2 the semi-infinite quadratic programming problem (4.1.16) can be reduced to an equivalent quadratic programming problem. Thus we are guaranteed to find a global solution.
Next we prove a stronger conclusion for the semi-infinite quadratic programming problem (4.1.16). We would need the following definitions and theorems from the approximation theory Cheney and Light (2009).

**Definition 3** (Positive Definite Function Cheney and Light (2009)). A function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be positive semi-definite if for any finite set of points $\{x_1, \ldots, x_m\}$, where $x_i \in \mathbb{R}^d$, the $m$-by-$m$ matrix $A$, where $A$ is defined such that its $(i,j)$-entry $A_{ij} = f(x_i - x_j)$, is positive semi-definite, i.e., for a vector $u \in \mathbb{R}^m$, $u^T Au \geq 0$. If $u^T Au > 0$ for any $u \neq 0$ and distinct set of points $x_i$, $i = 1, 2, \ldots, m$, then the function $f$ is said to be strictly positive definite.

**Definition 4** (Radial Cheney and Light (2009)). A real-valued function $f$ on an inner product space is said to be radial if $f(x) = f(y)$ whenever $\|x\| = \|y\|$.

**Theorem 6.** [Positive Definiteness of Gaussian Function Cheney and Light (2009)] If $\gamma > 0$, then the function $f(x) = e^{-\gamma \|x\|^2}$ is radial and strictly positive definite on any real inner product space.

Now we are ready to prove the following stronger conclusion.

**Theorem 7.** For distinct strike prices $\{x_1, \ldots, x_n\}$, where $x_i \geq 0$, $i = 1 \ldots n$, the semi-infinite quadratic programming problem (4.1.16) has a unique solution.

**Proof.** Recall that the semi-infinite quadratic programming problem (4.1.16) is

$$
\min_{\alpha_i, \xi_i} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j e^{-\gamma(x_i-x_j)^2} + \lambda \sum_{i=1}^{n} \xi_i^2
$$

s.t.

\[
\begin{align*}
-\varepsilon - \xi_i & \leq c_i - e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{x_i}^{\infty} (S - x_i) e^{-\gamma(S-x_i)^2} dS, & i = 1, \ldots, n. \\
c_i - e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{x_i}^{\infty} (S - x_i) e^{-\gamma(S-x_i)^2} dS & \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
\sum_{i=1}^{n} \alpha_i e^{-\gamma(x-x_i)^2} & \geq 0, & K \in [0, \infty) \\
\sum_{i=1}^{n} \alpha_i \left( \frac{1}{\sqrt{2\pi}} \left( \text{erf}(x_i\sqrt{\gamma}) + 1 \right) \right) & = 1 \\
e^{-rt} \sum_{i=1}^{n} \alpha_i \left( \frac{1}{\sqrt{\gamma}} \sqrt{\frac{\pi}{2}} e^{-\gamma x_i^2} + x_i \frac{\sqrt{\pi}}{\sqrt{2}} \left( \text{erf}(x_i\sqrt{\gamma}) + 1 \right) \right) & = S_0 e^{-\delta t}
\end{align*}
\]
The variables of the estimation problem are $\alpha_i, \xi_i$, $i = 1, \ldots, n$. As we can see that the objective function is quadratic in terms of these variables. And the constrains are linear in terms of these variables.

By theorem 2 the semi-infinite quadratic programming problem (4.1.16) can be reduced to an equivalent quadratic programming problem. Thus to prove the uniqueness of the solution, we only need to prove the quadratic term in the objective function in strictly convex, i.e., the involved matrix in the quadratic term is strictly positive definite.

Recall $\mathcal{K}(x_i, x) = e^{-\gamma(x-x_i)^2}$, $x \in [0, \infty)$. For distinct strike prices $\{x_1, \ldots, x_n\}$, the Kernel Matrix $M$ can be written as:

$$
M = 
\begin{bmatrix}
\mathcal{K}(x_1, x_1) & \mathcal{K}(x_1, x_2) & \mathcal{K}(x_1, x_3) & \cdots & \mathcal{K}(x_1, x_n) \\
\mathcal{K}(x_2, x_1) & \mathcal{K}(x_2, x_2) & \mathcal{K}(x_2, x_3) & \cdots & \mathcal{K}(x_2, x_n) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\mathcal{K}(x_n, x_1) & \mathcal{K}(x_n, x_2) & \mathcal{K}(x_n, x_3) & \cdots & \mathcal{K}(x_n, x_n)
\end{bmatrix}
$$

By theorem 6, $M$ is strictly positive definite, i.e., for any vector $v_1 \in \mathbb{R}^n$,

$$
v_1^T M v_1 \geq 0
$$

And $v_1^T M v_1 > 0$ if $v_1 \neq 0$.

The objective function can be written as:

$$
u^T A u = u^T \begin{bmatrix} \frac{1}{2} M & 0 \\ 0 & \lambda I \end{bmatrix} u = \frac{1}{2} u_1^T M u_1 + \lambda u_2^T I u_2 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j e^{-\gamma(x_i-x_j)^2} + \lambda \sum_{i=1}^{n} \xi_i^2
$$

where $u \in \mathbb{R}^{2n}$ and $u^T = \{u_1^T, u_2^T\} = \{\alpha_1, \alpha_2, \ldots, \alpha_n, \xi_1, \xi_2, \ldots, \xi_n\}$, $I$ is the $n$-by-$n$ identity matrix, $A$ is a $2n$-by-$2n$ matrix,

$$
A = \begin{bmatrix} \frac{1}{2} M & 0 \\ 0 & \lambda I \end{bmatrix}
$$
For any vector \( v \in R^{2n} \), we have
\[
v^T Av = [v_1^T, v_2^T] A \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = [v_1^T, v_2^T] \begin{bmatrix} \frac{1}{2} M & 0 \\ 0 & \lambda I \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \frac{1}{2} v_1^T M v_1 + \lambda v_2^T I v_2
\]

By the positive definiteness of \( M \), \( \frac{1}{2} v_1^T M v_1 \geq 0 \). And it is apparent that \( \lambda v_2^T I v_2 \geq 0 \).

If \( v \neq 0 \), then either \( v_1 \neq 0 \) or \( v_2 \neq 0 \). So one of \( \frac{1}{2} v_1^T M v_1 \geq 0 \), \( \lambda v_2^T I v_2 \geq 0 \) is going to be strict.

So for any \( v \in R^{2n} \),
\[
v^T Av = \frac{1}{2} v_1^T M v_1 + \lambda v_2^T I v_2 \geq 0
\]

And if \( v \neq 0 \),
\[
v^T Av = \frac{1}{2} v_1^T M v_1 + \lambda v_2^T I v_2 > 0
\]

So the matrix \( A \) in the objective function is strictly positive definite. Thus we are guaranteed to find a unique solution.

\( \lambda, \gamma, \varepsilon \) are three positive parameters we need to determine before solving the optimization problem (4.1.16). \( \lambda \) is the trade-off parameter between the flatness of the RND function and the goodness of fit of the data. \( \gamma \) is the scale parameter for the RBF kernel. \( \varepsilon \) is the tube size used in the \( \varepsilon \)-insensitive square loss function. We will use a data-driven method, i.e., the cross-validation method to determine these three parameters. Details are given in the later section.

### 4.2 Benchmark

In section 3.4 we talked about the comparison between SML and LPSVR. LPSVR outperforms SML in terms of both accuracy and stability, i.e., LPSVR has a smaller RISB and RIV, in total a smaller RIMSE.

Then in section 3.6 we compared the performance among SML, LPSVR and \( \varepsilon_i \)-LPSVR. \( \varepsilon_i \)-LPSVR also outperforms SML. Compared to LPSVR, \( \varepsilon_i \)-LPSVR improves the accuracy level with a smaller RISB. However, the stability level, i.e, RIVs of both LPSVR and \( \varepsilon_i \)-LPSVR are similar to each other. In general, \( \varepsilon_i \)-LPSVR has a relatively smaller RIMSE. Here we take both LPSVR and \( \varepsilon_i \)-LPSVR as benchmark methods to compare them with QPSVR.
Recall optimization problem 3.1.13 by LPSVR:

\[
\begin{align*}
\min_{\alpha_i, d_i, \xi_i} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad \begin{cases}
-\varepsilon - \xi_i \leq c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1+(S/x_i)^{\beta})^2} dS, & i = 1, \ldots, n. \\
c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1+(S/x_j)^{\beta})^2} dS \leq \varepsilon + \xi_i, & i = 1, \ldots, n. \\
-d_i \leq \alpha_i \leq d_i, & i = 1, \ldots, n. \\
\sum_{i=1}^{n} \alpha_i \geq 0, & K \in [0, \infty) \\
\sum_{i=1}^{n} \alpha_i = 1 \\
e^{-rt} \frac{\pi/\beta}{\sin(\pi/\beta)} \sum_{i=1}^{n} \alpha_i x_i = S_0 e^{-\delta t}
\end{cases}
\end{align*}
\]

Recall optimization problem 3.5.5 by \(\varepsilon_i\)-LPSVR:

\[
\begin{align*}
\min_{\alpha_i, d_i, \xi_i} \sum_{i=1}^{n} d_i + \lambda \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad \begin{cases}
-\varepsilon_i - \xi_i \leq c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1+(S/x_j)^{\beta})^2} dS, & i = 1, \ldots, n. \\
c_i - e^{-rt} \sum_{j=1}^{n} \alpha_j \int_{x_i}^{\infty} (S - x_i) \frac{(\beta/x_i)(S/x_i)^{\beta-1}}{(1+(S/x_j)^{\beta})^2} dS \leq \varepsilon_i + \xi_i, & i = 1, \ldots, n. \\
-d_i \leq \alpha_i \leq d_i, & i = 1, \ldots, n. \\
\sum_{i=1}^{n} \alpha_i \geq 0, & K \in [0, \infty) \\
\sum_{i=1}^{n} \alpha_i = 1 \\
e^{-rt} \frac{\pi/\beta}{\sin(\pi/\beta)} \sum_{i=1}^{n} \alpha_i x_i = S_0 e^{-\delta t}
\end{cases}
\end{align*}
\]

\[4.3 \text{ Monte-Carlo Simulations}\]

We follow the same Monte-Carlo simulations designed in section 3.3 so we can compare QPSVR with LPSVR and \(\varepsilon_i\)-LPSVR.

The implementation details for optimization problem 4.1.16 are similar to the process in section 3.3.3 except some minor modifications. We include them here for elaboration.
We have three extra parameters to determine before solving the optimization problem (4.1.16), the trade-off parameter between the flatness of the RND function and the goodness of fit of the data $\lambda$, the scale parameter for the RBF kernel $\gamma$, the tube size used in the $\varepsilon$-insensitive square loss function $\varepsilon$.

We start with a grid search and a data-driven method, i.e., the 10-fold cross-validation method is used to determine these three parameters. The whole process of solving the estimation problem for one of the cases is as follows:

- **Step 1**: Divide 61 data points into 10 groups with the first group containing 7 numbers and the rest containing 6 numbers. Within each group the strikes are equally spaced with the step size 250.

- **Step 2**: Determine the range of these three parameters for grid search. We use $2^i$, $i = 1, \ldots, 10$ for $\lambda$, $1e^{-5}$ to $1e^{-2}$ with step size $1e^{-5}$ for $\gamma$, $0$ to $\max(s_i)$, $i = 1, \ldots, 53$ with step size $0.01$ for $\varepsilon$ where $s_i$ is the option’s bid-ask spread at strike price $x_i$.

- **Step 3**: Start with a set of value of three parameters and use 9 groups of data to solve the optimization problem (4.1.16). Evaluate the absolute deviation of option prices on the rest group.

- **Step 4**: Take turns to leave a group of data out to solve the optimization problem (4.1.16) and evaluate the absolute deviation of option prices on that reserved group using the same set of parameters as Step 3. In total we solved the optimization problem 10 times.

- **Step 5**: Calculate the mean of the absolute deviation of option prices for the set of parameters.

- **Step 6**: Start with another set of value of three parameters and repeat from Step 3 to Step 6 until the grid search is finished.

- **Step 7**: Pick the set of parameters with the minimum mean absolute deviation of the option prices and resolve the optimization problem (4.1.16) with 10 groups of data to obtain the estimated RND.
• Step 8: Repeat Step 1 to Step 7 with 1000 different generated data sets and obtain 1000 estimated RNDs.

We repeat the whole process for four cases and after that, we have obtained 1000 estimated RNDs for the real RND in each case.

4.4 Measurements and Results

We use $RIMSE$, $RISB$ and $RIV$ explained in section 3.4 as measurements to compare the performance of QPSVR and benchmark methods.

\[
RIMSE = \left( \int_0^\infty E[(\hat{f}(x) - f(x))^2] dx \right)^{1/2}
\]

\[
RISB = \left( \int_0^\infty [E(\hat{f}(x) - f(x))^2] dx \right)^{1/2}
\]

\[
RIV = \left( \int_0^\infty E[(\hat{f}(x) - E\hat{f}(x))^2] dx \right)^{1/2}
\]

where $\hat{f}(x)$ is the estimated RND and $f(x)$ is the real RND.

The fitting results of the optimization problem (4.1.16) are shown in Figure 4.1, 4.2, 4.3, 4.4, where we compare the performance between $\varepsilon_i$-LPSVR and QPSVR by showing the fitted RND graphically.

The measurement results are shown in Table 4.1.

To better interpret Table 4.1, Figure 4.5, 4.6, 4.7, 4.8 are generated.

Compared with LPSVR, QPSVR improves apparently in RIV and maintains a similar level in RISB and thus in total a smaller RIMSE. Generally QPSVR outperforms LPSVR with an improvement in the stability level and a close performance in the accuracy level.

Compared with $\varepsilon_i$-LPSVR, QPSVR also improves apparently in RIV, but scores larger in RISB. In total QPSVR has a smaller RIMSE so the RIV takes a bigger part in the RIMSE than the RISB. Generally QPSVR outperforms $\varepsilon_i$-LPSVR with a big improvement in the stability level and a relatively poor performance in the accuracy level.

In total QPSVR outperforms LPSVR and $\varepsilon_i$-LPSVR.
Table 4.1: The RIMSEs, RISBs and RIVs of LPSVR, $\varepsilon_i$-LPSVR and QPSVR

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPSVR</td>
<td>RIMSE</td>
<td>0.1579</td>
<td>0.1428</td>
<td>0.1378</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0425</td>
<td>0.0747</td>
<td>0.0563</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.1521</td>
<td>0.1217</td>
<td>0.1258</td>
</tr>
<tr>
<td>$\varepsilon_i$-LPSVR</td>
<td>RIMSE</td>
<td>0.1396</td>
<td>0.1283</td>
<td>0.1178</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0303</td>
<td>0.0386</td>
<td>0.0364</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.1363</td>
<td>0.1224</td>
<td>0.1120</td>
</tr>
<tr>
<td>QPSVR</td>
<td>RIMSE</td>
<td>0.1141</td>
<td>0.1071</td>
<td>0.1097</td>
</tr>
<tr>
<td></td>
<td>RISB</td>
<td>0.0388</td>
<td>0.0659</td>
<td>0.0472</td>
</tr>
<tr>
<td></td>
<td>RIV</td>
<td>0.1073</td>
<td>0.0844</td>
<td>0.0990</td>
</tr>
</tbody>
</table>

4.5 $\varepsilon_i$-QPSVR

We name the proposed scheme in this section as $\varepsilon_i$-QPSVR.

In section 4.1 we formulate the estimation of the RND into a semi-infinite quadratic programming problem with the $\varepsilon$-insensitive square loss function.

$$|\xi_i|^2_{e_i} := \begin{cases} 0, & \text{if } |\xi_i| \leq \varepsilon_i \\ (|\xi_i| - \varepsilon_i)^2, & \text{otherwise} \end{cases}$$

The idea of this loss function is to set a common tube size, penalize the points outside the tube and neglect points inside the tube. The penalty is counted in a square form, i.e., the square of the distance from the outside point to the closest boundary of the tube.

The advantage of the loss function is the sparsity brought by the tube. The tube size is determined by grid search and cross validation in section 4.3. Inspired by the standard $\varepsilon$-insensitive square loss function and the idea explained in section 3.5, we define the following $\varepsilon$-insensitive square loss function:

$$|\xi_i|^2_{e_i} := \begin{cases} 0, & \text{if } |\xi_i| \leq \varepsilon_i \\ (|\xi_i| - \varepsilon_i)^2, & \text{otherwise} \end{cases}$$

$$\varepsilon_i := vS_i$$
where $S_i$ is the bid-ask spread at strike price $x_i$ and we restrict the proportion parameter $v \in (0, 1/2)$. If $v = 0$, then we penalize each data point by the $L_2$ norm. If $v = 1/2$, then we set the threshold for each data point as half of its bid-ask spread, meaning that we allow it to deviate up or down from the observed option price by half of its spread and we count the penalty as the squared term.

By setting $\varepsilon_i := vS_i$, the penalty threshold is closely related to the noise level. By searching for the proportion parameter $v$, instead of a common tube size $\varepsilon$, we gain more flexibility within the same amount of computation. Besides, we obtain variable penalty thresholds while avoiding exhaustively searching for arbitrary shapes.

The objective function is:

$$
\frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} |c_i - C(x_i)|^2_{\varepsilon_i}
$$

(4.5.1)

For a specified $v$, i.e., specified $\varepsilon_i$, $i = 1, 2, \ldots, n$, incorporating no-arbitrage and the RND constraints from equation (2.1.15), we can formulate the estimation problem as:

$$
\min_{b, \alpha_i, \xi_i, \xi^*_i} \quad \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} (\xi_i^2 + (\xi^*_i)^2)
$$

s.t.

$$
\begin{cases}
\xi_i - C(x_i) \leq \varepsilon_i + \xi_i, & i = 1, \ldots, n. \\
C(x_i) - \xi_i \leq \varepsilon_i + \xi^*_i, & i = 1, \ldots, n. \\
\xi_i, \xi^*_i \geq 0, & i = 1, \ldots, n. \\
f(K) \geq 0, & K \in [0, \infty) \\
f_0^{\infty} f(S)dS = 1 \\
C(0) = S_0e^{-\delta t}
\end{cases}
$$

(4.5.2)
As in section 4.1, the problem can be rewritten as:

\[
\begin{align*}
\min_{b, \alpha_i, \xi_i} & \quad \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) + \lambda \sum_{i=1}^{n} \xi_i^2 \\
\text{s.t.} & \quad -\varepsilon_i - \xi_i \leq c_i - C(x_i) \leq \varepsilon_i + \xi_i, \quad i = 1, \ldots, n. \\
& \quad f(K) \geq 0, \quad K \in [0, \infty) \\
& \quad \int_{0}^{\infty} f(S)dS = 1 \\
& \quad C(0) = S_0 e^{-\delta t}
\end{align*}
\]  \tag{4.5.3}

The first constraint implies \( \xi_i \geq 0, \quad i = 1, \ldots, n. \)

We set \( b = 0 \) for simplicity and choose the RBF kernel:

\[ K(x_i, x) = \exp(-\gamma \|x - x_i\|^2) \]  \tag{4.5.4}

where \( \gamma > 0 \) is the scale parameter.

By a very similar argument as in theorem 5 and 7, we can prove the following theorems for the optimization problem (4.5.3).

**Theorem 8.** For the RBF kernel: \( K(x_i, x) = \exp(-\gamma \|x - x_i\|^2) \), \( x \in [0, \infty) \), and \( b = 0 \), the optimization problem (4.5.3) has a global solution.

**Proof.** Replace \( \varepsilon \) by \( \varepsilon_i \) in the optimization problem 4.1.16 and by the same argument as in theorem 5, we can prove theorem 8. \( \square \)

The optimization problem (4.5.3) finally becomes:

\[
\begin{align*}
\min_{\alpha_i, \xi_i} & \quad \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j e^{-\gamma(x_i-x_j)^2} + \lambda \sum_{i=1}^{n} \xi_i^2 \\
\text{s.t.} & \quad -\varepsilon_i - \xi_i \leq c_i - e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{x_i}^{\infty} (S - x_i) e^{-\gamma(S-x_i)^2} dS, \quad i = 1, \ldots, n. \\
& \quad c_i - e^{-rt} \sum_{i=1}^{n} \alpha_i \int_{x_i}^{\infty} (S - x_i) e^{-\gamma(S-x_i)^2} dS \leq \varepsilon_i + \xi_i, \quad i = 1, \ldots, n. \\
& \quad \sum_{i=1}^{n} \alpha_i e^{-\gamma(x-x_i)^2} \geq 0, \quad K \in [0, \infty) \\
& \quad \sum_{i=1}^{n} \alpha_i \left( \frac{1}{\sqrt{\gamma}} \sqrt{\pi} \left( \text{erf}(x_i \sqrt{\gamma}) + 1 \right) \right) = 1 \\
& \quad e^{-rt} \sum_{i=1}^{n} \alpha_i \left( \frac{1}{2\sqrt{\gamma}} e^{-\gamma x_i^2} + \frac{x_i}{\sqrt{\gamma}} \sqrt{\pi} \left( \text{erf}(x_i \sqrt{\gamma}) + 1 \right) \right) = S_0 e^{-\delta t}
\end{align*}
\]  \tag{4.5.5}
The variables of the estimation problem are \( \alpha_i, \xi_i, i = 1, \ldots, n \). As we can see that the objective function is quadratic in terms of these variables. And the constrains are linear in terms of these variables. In the third constraint we have a continuous variable \( K \) which results in infinite inequalities. So it is also a semi-infinite quadratic programming problem.

**Theorem 9.** For distinct strike prices \( \{x_1, \ldots, x_n\} \) where \( x_i \geq 0, i = 1 \ldots n \), the semi-infinite quadratic programming problem (4.5.5) has a unique solution.

*Proof.* Replace \( \varepsilon \) by \( \varepsilon_i \) in the optimization problem 4.5.5 and by the same argument as in theorem 7, we can prove theorem 9.

\( \lambda, \gamma, v \) are three positive parameters we need to determine before solving the optimization problem (4.5.5). \( \lambda \) is the trade-off parameter between the flatness of the RND function and the goodness of fit of the data. \( \gamma \) is the scale parameter for the RBF kernel. \( v \) is the proportion parameter used in the \( \varepsilon_i \)-insensitive square loss function, where \( \varepsilon_i := vS_i \). We will use a data-driven method, i.e., the cross-validation method to determine these three parameters. Details are given in the later section.

### 4.6 Simulation Results

The simulation process for the optimization problem (4.5.5) is the same for the optimization problem (4.1.16). We are using the same generated data here so we can compare the performance among LPSVR, \( \varepsilon \)-LPSVR, QPSVR and \( \varepsilon \)-QPSVR.

There is one parameter that needs to be addressed separately, i.e., the proportion parameter \( v \). We just replace the search for \( \varepsilon \) by the search for \( v \) using \( \frac{1}{n} \), where \( n \) is an integer and \( n = 2, \ldots, 100 \).

The fitting results of the optimization problem (4.5.5) are shown in Figure 4.9, 4.10, 4.11, 4.12, where we compare the performance between QPSVR and \( \varepsilon \)-QPSVR by showing the fitted RND graphically.

The measurement results are shown in Table 4.2.

To better interpret Table 4.2, Figure 4.13, 4.14 , 4.15, 4.16 are generated.
Table 4.2: The RIMSEs, RISBs and RIVs of LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPSVR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$RIMSE$</td>
<td>0.1579</td>
<td>0.1428</td>
<td>0.1378</td>
<td>0.1540</td>
</tr>
<tr>
<td>$RISB$</td>
<td>0.0425</td>
<td>0.0747</td>
<td>0.0563</td>
<td>0.0706</td>
</tr>
<tr>
<td>$RIV$</td>
<td>0.1521</td>
<td>0.1217</td>
<td>0.1258</td>
<td>0.1369</td>
</tr>
<tr>
<td>$\varepsilon_i$-LPSVR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$RIMSE$</td>
<td>0.1396</td>
<td>0.1283</td>
<td>0.1178</td>
<td>0.1433</td>
</tr>
<tr>
<td>$RISB$</td>
<td>0.0303</td>
<td>0.0386</td>
<td>0.0364</td>
<td>0.0412</td>
</tr>
<tr>
<td>$RIV$</td>
<td>0.1363</td>
<td>0.1224</td>
<td>0.1120</td>
<td>0.1373</td>
</tr>
<tr>
<td>QPSVR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$RIMSE$</td>
<td>0.1141</td>
<td>0.1071</td>
<td>0.1097</td>
<td>0.1172</td>
</tr>
<tr>
<td>$RISB$</td>
<td>0.0388</td>
<td>0.0659</td>
<td>0.0472</td>
<td>0.0769</td>
</tr>
<tr>
<td>$RIV$</td>
<td>0.1073</td>
<td>0.0844</td>
<td>0.0990</td>
<td>0.0884</td>
</tr>
<tr>
<td>$\varepsilon_i$-QPSVR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$RIMSE$</td>
<td>0.0996</td>
<td>0.0879</td>
<td>0.0822</td>
<td>0.1083</td>
</tr>
<tr>
<td>$RISB$</td>
<td>0.0284</td>
<td>0.0426</td>
<td>0.0377</td>
<td>0.0429</td>
</tr>
<tr>
<td>$RIV$</td>
<td>0.0955</td>
<td>0.0769</td>
<td>0.0731</td>
<td>0.0994</td>
</tr>
</tbody>
</table>

Compared with LPSVR, $\varepsilon_i$-QPSVR outperforms in both accuracy and stability with smaller RISB and RIV and in total a smaller RIMSE.

Compared with $\varepsilon_i$-LPSVR, $\varepsilon_i$-QPSVR improves apparently in RIV and maintains a similar level in RISB and thus in total a smaller RIMSE. Generally $\varepsilon_i$-QPSVR outperforms $\varepsilon_i$-LPSVR with an improvement in the stability level and a close performance in the accuracy level.

Compared with QPSVR, $\varepsilon_i$-QPSVR improves apparently in RISB and maintains a similar level in RIV and thus in total a smaller RIMSE. Generally $\varepsilon_i$-QPSVR outperforms QPSVR with an improvement in the accuracy level and a close performance in the stability level.

In total, $\varepsilon_i$-QPSVR is the best one among LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR in terms of both accuracy and stability.
Figure 4.1: Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR.

Figure 4.2: Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR.
Figure 4.3: Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR.

Figure 4.4: Estimated RND of S&P 500 index option data by $\varepsilon_i$-LPSVR (left) and QPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by $\varepsilon_i$-LPSVR or QPSVR.
Figure 4.5: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 1.

Figure 4.6: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 2.
Figure 4.7: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 3.

Figure 4.8: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR and QPSVR for Case 4.
Figure 4.9: Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 1. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR.

Figure 4.10: Estimated RND of S&P 500 index option data by QPSVR (left) and $\varepsilon_i$-QPSVR (right) for Case 2. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or $\varepsilon_i$-QPSVR.
Figure 4.11: Estimated RND of S&P 500 index option data by QPSVR (left) and \( \varepsilon_i \)-QPSVR (right) for Case 3. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or \( \varepsilon_i \)-QPSVR.

Figure 4.12: Estimated RND of S&P 500 index option data by QPSVR (left) and \( \varepsilon_i \)-QPSVR (right) for Case 4. 'Real' is the constructed underlying real RND. 'Fit' is the mean of the estimated 1000 RNDs by QPSVR or \( \varepsilon_i \)-QPSVR.
Figure 4.13: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_1$-LPSVR, QPSVR and $\varepsilon_1$-QPSVR for Case 1.

Figure 4.14: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_1$-LPSVR, QPSVR and $\varepsilon_1$-QPSVR for Case 2.
Figure 4.15: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 3.

Figure 4.16: The RIMSEs, RISBs and RIVs of estimated RNDs by LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR for Case 4.
CHAPTER 5. SUMMARY AND DISCUSSION

In this dissertation, we propose non-parametric methods to estimate the RND from European call option prices in the market. The method is based on SVR and together with the involved constraints we formulate the estimation process into optimization problems.

We develop the methods in terms of Linear Programming and Quadratic Programming.

Under the framework of Linear Programming, we propose a method called LPSVR using the standard $\varepsilon$-insensitive loss function which formulates the estimation process into a semi-infinite linear programming problem. Global solution is guaranteed by this scheme. The performance of LPSVR is evaluated by Monte-Carlo simulations. Compared with the benchmark SML method, the proposed method achieves better accuracy and stability.

We propose another method under this framework called $\varepsilon_i$-LPSVR using $\varepsilon_i$-insensitive loss function which also formulates the estimation process into a semi-infinite linear programming problem. This method uses a vary tube size instead of a common one which essentially changes the penalty scheme. Global solution is also guaranteed by this scheme. Compared with SML, $\varepsilon_i$-LPSVR outperforms both in accuracy and stability. Compared with LPSVR, $\varepsilon_i$-LPSVR improves the accuracy level and maintains the stability level. In total, $\varepsilon_i$-LPSVR outperforms LPSVR.

Under the framework of Quadratic Programming, we propose a method called QPSVR using the modified $\varepsilon$-insensitive square loss function which formulates the estimation process into a semi-infinite quadratic programming problem. Globalness and uniqueness of the solution are guaranteed by this scheme. The performance of QPSVR is evaluated by Monte-Carlo simulations. Compared with LPSVR, QPSVR improves apparently in RIV and maintains the level in RISB and thus in total a smaller RIMSE. Compared with $\varepsilon_i$-LPSVR, QPSVR also improves apparently in RIV, but scores larger in RISB. In total QPSVR has a smaller RIMSE so the RIV takes a bigger part in the RIMSE than the RISB. In total QPSVR outperforms LPSVR and $\varepsilon_i$-LPSVR.
We propose another method under this framework called $\varepsilon_i$-QPSVR using $\varepsilon_i$-insensitive square loss function which also formulates the estimation process into a semi-infinite quadratic programming problem. This method uses a vary tube size instead of a common one which essentially changes the penalty scheme. Globalness and uniqueness of the solution are also guaranteed by this scheme.

Compared with LPSVR, $\varepsilon_i$-QPSVR outperforms in both accuracy and stability with smaller RISB and RIV and in total a smaller RIMSE. Compared with $\varepsilon_i$-LPSVR, $\varepsilon_i$-QPSVR improves apparently in RIV and maintains the level in RISB and thus in total a smaller RIMSE. Compared with QPSVR, $\varepsilon_i$-QPSVR improves apparently in RISB and maintains the level in RIV and thus in total a smaller RIMSE. In total, $\varepsilon_i$-QPSVR is the best one among LPSVR, $\varepsilon_i$-LPSVR, QPSVR and $\varepsilon_i$-QPSVR in terms of both accuracy and stability.

The future work can be about the investigation of different kernels or the search for other statistical learning methods. Since we use one type of kernel within one scheme, researchers can investigate multiple kernels learning methods which may or may not satisfy the Mercer Condition. Other statistical learning methods such as neural networking may provide another alternative scheme and open another point of view.


Ian, I. and Choo, E. (n.d.). Practical option pricing with support vector regression and mart.


