Data-Driven Models: An Alternative Discrete Hedging Strategy

by

Ke Nian

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	undetermined Professor, School of Computer Science, University of Undetermined
Supervisor(s):	Yuying Li Professor, School of Computer Science, University of Waterloo Thomas F. Coleman Professor, Department of Combinatoric and Optimization, University of Waterloo
Internal Examiner:	Justin W.L. Wan Professor, School of Computer Science, University of Waterloo
Internal Examiner:	Yaoliang Yu Assistant Professor, School of Computer Science, University of Waterloo
Internal-External Member:	undetermined Professor, Dept. of Math of Science, University of Waterloo

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Options hedging is a critical problem in financial risk management. The prevailing approach in financial derivative pricing and hedging has been to firstly assume a parametric model describing the underlying price dynamics. An option model function V is then calibrated to current market option prices and various sensitivities are computed and used to hedge the option risk. It has been recognized that computing hedging position from the sensitivity of the calibrated model option value function is inadequate in minimizing the variance of the option hedging risk, as it fails to capture the model parameter dependence on the underlying price. We propose several data-driven approaches to directly learn a hedging function from the historical market option and underlying data by minimizing certain measure of the local hedging risk and total hedging risk. This thesis will focus on answering the following questions: 1) Can we efficiently build direct data-driven models for discrete hedging problem that outperform existing state-of-art parametric hedging models based on the market prices? 2) Can we incorporate feature selection and feature extraction into the data-driven models to further improve the performance of the discrete hedging? 3) Can we build efficient models for both the one-step local risk hedging problem and multi-steps total risk hedging problem based on the state-of-art learning framework such as deep learning framework and kernel learning framework?

Using the S&P 500 index daily option data for more than a decade ending in August 2015, we firstly propose a direct data-driven approach [143] based on kernel learning framework and we demonstrate that the proposed method outperforms the parametric minimum variance hedging method proposed in [112], as well as minimum variance hedging corrective techniques based on stochastic volatility or local volatility models. Furthermore, we show that the proposed approach achieves significant gain over the implied Black-Sholes delta hedging for weekly and monthly hedging.

Following the direct data-driven kernel learning approach [143], we propose a robust encoder-decoder Gated Recurrent Unit (GRU) model, GRU_δ, for optimal discrete option hedging. The proposed GRU_δ utilizes the Black-Scholes model as a pre-trained model and incorporates sequential information and feature selection. Using the S&P 500 index European option market data from January 2, 2004 to August 31, 2015, we demonstrate that the weekly and monthly hedging performance of the proposed GRU_δ significantly surpasses that of the data-driven minimum variance (MV) method in [112], the regularized kernel data-driven model [143], and the SABR-Bartlett method [98]. In addition, the daily hedging performance of the proposed GRU_δ also surpasses that of MV methods in [112] based on parametric models, the kernel method [143] and SABR-Bartlett method [98].

Lastly, we design a multi-steps data-driven models $\text{GRU}_{\text{TOTAL}}$ based on the GRU_δ to hedge the option discretely until the expiry. We utilize SABR model and Local Volatility Function (LVF) to augment existing market data and thus alleviate the problem of scarcity in market option prices. The augmented market data is used to train a sufficient total risk hedging model $\text{GRU}_{\text{TOTAL}}$.

Acknowledgements

I would like to thank all the people who made this thesis possible.

Dedication

This is dedicated to the one I love.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

One of the critical problems in financial risk management is to hedge the risk of trading option. The traditional approach in financial derivative pricing and hedging relies heavily on parametric assumptions describing the dynamics of underlying asset. The common practice is to calibrate an option pricing function based on the specific parametric model and compute various sensitivities to hedge option risk. For example, the sensitivity of the option value function to the underlying price is used in delta hedging. Under the assumption of a complete market [165] where one can continuously rebalance the hedging portfolio, the value of an option written on the underlying asset can be perfectly replicated by a hedging portfolio consisting of the underlying asset and the risk-free asset. In practice, we have to rebalance the hedging portfolio discretely instead of continuously due to the existence of the transaction cost. The practice of adjusting the hedging portfolio discretely is often referred to as discrete hedging.

There are many parametric models proposed to describe the dynamics of underlying asset. The original and most celebrated parametric Black-Scholes (BS) model uses a constant volatility [20, 139], which is shown to produce inaccurate option prices particularly for deeply out-of-the-money options and deeply in-the-money options [87]. In addition, the BS model is unable to capture the non-zero correlation between the volatility and the underlying asset price, e.g., [80, 22]. The practitioner's BS delta hedging approach sets the constant volatility in the BS model to the implied volatility calibrated to the market price at the time of re-balancing. Many alternative parametric models have been proposed to improve the BS model, including the Stochastic Volatility (SV) model, e.g., [99, 103, 111, 13], the Local Volatility Function (LVF) model [45, 66, 158, 68], and the jump model, e.g., [100, 122]. Unfortunately, all models have been shown to have their limitations in accurately modeling option market prices.

Errors in the option value model have significant implications in hedging. Consider, for example, when the hedging position is computed from the sensitivity of the option value function calibrated at the hedging time, the computed hedging position only depends on the assumed dynamics of underlying price and the current market option prices. Unless the assumed dynamics for the underlying price is exact and all assumptions that results in the option pricing function are all valid, the option function calibrated at the hedging time cannot predict the future option

market price.

Specifically, let $V(S,t,T,K,r,q;\theta^*)$ be the option value function and θ^* be the vector of model parameters of the assumed option pricing model. At the hedging time t, assume that we calibrate the model price to match the market option price so that

$$V(S_t,t,T,K,r_t,q_t;\theta^*) = V^{mkt}_{t,T,K} \tag{1.0.1}$$

where $V^{mkt}_{t,T,K}$ denotes the actual market option price at time t with strike price K and the expiry date T, S_t denotes the underlying price at t, r_t denotes the risk-free rate at t and q_t denotes the dividend yield at t. The option value function $V(S,t,T,K,r,q;\theta^*)$, calibrated to market price at time t, does not ensure that the option delta from the pricing model $\frac{\partial V}{\partial S}$ equals to $\frac{\partial V^{mkt}}{\partial S}$, which is indeed unknown, and requires the modelling of the dependence of the calibrated model parameter on the underlying price [143, 45, 112]. The missing sensitivity $\frac{\partial \theta^*}{\partial S}$, is difficult to account for and is often ignored, though for some models, corrections have been proposed to capture the dependence [112, 98, 16].

Since machine learning algorithms usually do not impose assumptions on the model to be learned, they have recently been adopted to determine an option value function directly from the market data, with the goal of avoiding the model misspecification issues from the parametric modeling approach e.g.,[93, 82, 114]. Unfortunately, using nonparametric learning, hedging positions still need to be computed from the sensitivity of the model value function. While no assumption is explicitly made for the dynamics of underlying asset, the option value function is determined by data through cross-validation, leading to training errors. Since there is no assurance that the sensitivity of the learned option value function with regards to underlying asset matches that the sensitivity of the market option price, the parameters of the model learned directly from data can similarly exhibit dependence on the underlying price. When the hedging position is computed from the partial derivative of the data-driven option value function, e.g., [114], this dependence cannot be accounted for and again is ignored. Hence, option hedging risk remains insufficiently minimized. Furthermore, many data-driven option pricing models previously proposed [93, 82, 114] face the challenges in avoiding arbitrage in the resulting pricing surfaces. Therefore, recent research [30, 192, 127] on data-driven pricing model often focuses on using machine learning techniques in modelling and predicting the implied volatility surface and then use the resulting implied volatilities with Black-Scholes model to ensure the absence of the arbitrage. Again, the dependence of the implied volatility on the underlying asset is still omitted in those work.

Furthermore, using option delta from pricing model $\frac{\partial V}{\partial S}$ as the hedging position becomes inadequate when discrete hedging is performed in practice, particularly when rebalancing becomes infrequent. Instead, optimal discrete hedging strategy can be determined directly using an appropriate objective in the discrete hedging context, e.g., minimizing the variance of the hedging error, [112, 8, 92].

In hedging, the ultimate goal is to discover a hedging strategy which minimizes the hedging error measured by the market option and underlying prices. With the increasing availability of

market option prices, a timely question arises: is it possible to learn optimal hedging positions directly from market option prices and underlying prices data? Up until now, research in learning the hedging position directly from market data is scarce. Recently, a data-driven approach [112] is proposed to learn a parametric model for the minimum variance delta hedging based on the analysis of the BS option greeks and underlying market prices. However, the proposed parametric model focuses on the instantaneous hedging error analysis in a parametric model framework.

In this thesis, we study the discrete option hedging problem by explicitly focusing on the issues arising from model specification errors and calibrated model parameter dependence on underlying. We illustrate that the inability to minimize variance of the hedging error, when determining hedging position from option value function from a parametric model, is also shared by an option model estimated from a nonparametric method. Although a nonparametric modeling approach to option value can potentially lead to smaller mis-specification error, we illustrate that non-parametric model parameters can similarly depend on the underlying. Consequently the sensitivity of the estimated option value function will not lead to the minimization of option hedging risk. Furthermore, the estimated pricing function inevitably has errors, due to both model mis-specification, discretization, and numerical roundoff errors. The error in the value function can potentially be substantially magnified in computing partial derivatives as hedging positions.

We explore several direct market data-driven approaches to bypass the challenges mentioned above to achieve effective hedging performance. We first propose a data-driven kernel learning approach [143] to learn a local risk minimization hedging model directly from the market data observed at the hedging time t. We learn a hedging function from the market data by minimizing the empirical local hedging risk with a suitable regularization. The local risk corresponds directly to the variance of the hedging error in the discrete rebalancing period. A novel encoder-decoder RNN model GRU_{δ} [144], to extract both sequential and local features at hedging time t from market prices, is also proposed to learn option hedging positions directly from the market. We include a feature weighting procedure to select the most relevant local features at hedging time t and sequential features for the sequential data-driven model GRU_{δ}. Lastly, in order to deal with multi-steps discrete total hedging scenarios where we hedge until the expiry of the option [145], we extend our sequential local hedging model GRU_{δ} to be $\text{GRU}_{\text{TOTAL}}$. We compare our data-driven approaches with the parametric approaches and demonstrate the effectiveness of the data-driven hedging models in terms of both local hedging risk and total hedging risk.

1.1 Contribution

The contributions of this thesis with respect to the data-driven kernel hedging model [143] are summarized below:

- We analyze and discuss the implications from model mis-specification in the option value function for discrete option hedging. We illustrate challenges in accounting for the depen-

dence of the calibrated model parameters on the underlying, which arises due to model mis-specification.

- We analyze a regularized kernel network for option value estimation and illustrate that the partial derivative of the estimated value function with respect to the underlying similarly does not minimize the variance of the local hedging risk in general, even not infinitesimally.

- We propose a data-driven approach to learn a hedging position function directly by minimizing the variance of the local hedging risk. Specifically we implement a regularized spline kernel method $\mathrm{DKL}_{\mathrm{SPL}}$ to nonparametrically estimate the hedging function from the market data.

- Using synthetic data sets, we compare daily, weekly, and monthly hedging performance using the kernel direct data-driven hedging approach with the performance of the indirect approach where hedging positions are computed from the sensitivity of the nonparametric option value function. In particular, we present computational results which demonstrate that the direct spline kernel hedging position learning outperforms the hedging position computed from the sensitivity of the spline kernel option value function.

- Using S&P 500 index option market data for more than a decade ending in August 31, 2015, we demonstrate that the daily hedging performance of the direct spline kernel hedging function learning method surpasses that of the minimum variance quadratic hedging formula proposed in [112], as well as corrective methods based on LVF and SABR implemented in [112].

- We also present weekly and monthly hedging results using the S&P 500 index option market data and demonstrate significant enhanced performance over the BS implied volatility hedging.

The contributions with respect to the data-driven hedging model with sequential features [144] are summarized below:

- We propose a novel encoder-decoder RNN model, to extract both sequential and current features from market prices, and to learn option hedging positions directly from the market. We include a feature weighting procedure to select the most relevant local features and sequential time series features for the data-driven model.

- To ensure robust learning, we use the Huber loss function as the learning objective, adaptively setting the error resolution parameter to the Black–Scholes hedging error, allowing it to varying from data instance to data instance. Furthermore, the proposed GRU_δ can be updated more frequently than the data-driven model in [143] to account for the market shifts.

- Using the S&P 500 index option market data from January 2, 2004 to August 31st, 2015, we demonstrate that the weekly and monthly hedging performance of the proposed GRU_δ

significantly surpasses that of the data-driven minimum variance (MV) method in [112], the regularized kernel data-driven model [143], and the SABR-Bartlett method [16].

- Using the S&P 500 index option market data from January 2, 2004 to August 31st, 2015, we demonstrate that the daily hedging performance of the proposed GRU_δ surpasses that of the minimum variance quadratic hedging method proposed in [112], the corrective methods based on LVF and SABR implemented in [112], the SABR-Bartlett method [16], as well as the data-driven model in [143].

- To motivate the roles of each major component of the proposed GRU_δ, we demonstrate performance sensitivity through computational experiments. In addition, we illustrate and analyze the relative importance of selected features.

The contributions with respect to the data-driven total hedging model [145] are summarized below:

- We extend the data-driven local hedging model GRU_δ [144] to total risk hedging where we rebalance multiple times until the expiries of the options.

- We augment the market data using SABR model and local volatility model to cope with the challenges of scarcity in market option data.

- Using the S&P 500 index call option market data from January 1, 2000 to August 31st, 2015, we demonstrate the effectiveness on weekly and monthly hedging performance of the proposed total hedging model $\text{GRU}_{\text{TOTAL}}$. We compare $\text{GRU}_{\text{TOTAL}}$ with the sequential data-driven local hedging model $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$, which adopts the same model structure but is trained with a different objective functions, the BS delta hedging model and the SABR-Bartlett method [16]. The hedging performance is evaluated on the expiries of options.

1.2 Outline

The remainder of the thesis is organized as follows. Chapter 2 reviews the existing derivative pricing models, discrete hedging problems, local and total hedging risk and various existing parametric approaches to hedge options. Chapter 3 discusses the kernel learning framework and introduces the data-driven kernel local risk hedging model DKL_{SPL}. Chapter 4 discusses the Recurrent Neural Network(RNN) framework and introduces the data-driven sequential local hedging model GRU_δ. Chapter 5 discusses the empirical results from the data-driven sequential local hedging model GRU_δ and data-driven kernel local hedging model DKL_{SPL}. Chapter 6 introduces the data-driven total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and presents the empirical comparisons between local risk hedging model $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$ and total risk hedging model $\text{GRU}_{\text{TOTAL}}$. Chapter 6 also discusses the challenges of using market data to build data-driven total risk hedging models and the data augmentation procedure to cope with the challenges. We conclude in Chapter 7 with summary remarks and potential extensions.

Chapter 2

Option Hedging: From Past to Present

In this chapter, we review the existing option pricing models and discuss the problem of pricing model parameters dependence on underlying asset. In addition, we specify the discrete hedging problem and define the total and local hedging risk. Most of the discussion in this chapter are drawn from [165, 103, 16, 98, 99].

2.1 Discrete Hedging Problem

A European style call or put option gives its buyer the right to buy or sell the underlying asset on the option expiry with a strike price. Let the strike price be K and the S_T be the underlying price at expiry T. The payoff of call options is:

$$\max(S_T - K, 0).$$

The payoff of put options is:

$$\max(K - S_T, 0).$$

When a market is complete, e.g., under the assumptions of the Black–Scholes framework, the option payoff can be perfectly replicated by continuously trading the underlying asset and a risk-free asset account. However, markets are incomplete in practice and the risk associated with options cannot be eliminated completely. On the other hand, reducing risk as much as possible remains the main goal of hedging.

In an incomplete market, risk minimization is not completely defined until one specifies how to measure risk [78, 79, 163, 69]. For European options, a pricing measure can be determined through quadratic risk minimization, see, e.g., [164, 163, 50]. In this framework risk is measured by the expected quadratic difference between the payoff of an option and the value of a self-financing hedging portfolio at expiry date. This is the key idea behind total risk minimization. However, this strategy may not always exist and may be difficult to compute, particularly under more complex asset price dynamics[48, 49].

An alternative to total risk minimization is local risk minimization. In contrast to total risk minimization, in local risk minimization the expected quadratic incremental cost is minimized. As can be seen for European options [48] and American options [49] , optimal local risk minimization hedging strategies typically lead to a small total risk. Moreover, it has been shown that the choice of measure for incremental cost is important when the market is highly incomplete, e.g., [48, 49].

Literature in risk minimization pricing and hedging has focused on assuming a parametric form of the underlying asset dynamics. For example, the underlying asset price is assumed to follow a geometric Brownian motion:

$$dS_t = \mu S_t dt + \sigma S_t dW_t.$$

and the derivation of the hedging position can be computed with a binomial model [48, 49]. The key distinction between our work and the previous work on local and total risk minimization is that we have no assumption on the underlying asset price dynamics and therefore our models are not bound with a specific parametric form.

In this thesis, we propose to learn discrete data-driven hedging positions for standard european options, calls or puts, based on observations of the option market prices on the same underlying price at a set of trading times. The goal of the data-driven discrete option hedging in this thesis is to learn a hedging position function δ in the underlying dynamically to minimize certain appropriate measurements of the hedging risk on the historical market data and then apply the model on the unobserved testing cases. We are interested in two different types of hedging risk:

- Local hedging risk
- Total hedging risk

Note that in this thesis, the empirical performance of an option hedging method are measured by real market option and underlying prices. The definition of the local hedging risk and total hedging risk in discrete hedging are explained in the following subsections.

2.1.1 Local Hedging Risk

Let Δt denote a fixed time interval. Each observation of a market option price $V^{mkt}_{t,T,K}$ is uniquely associated with a triplet $\{t,T,K\}$, where t is the trading time of the option price, K is the strike, and expiry T. Furthermore, we assume the risk-free interest rate $r = 0$ in the discussion in this section. Denote:

$$\begin{aligned} \Delta V^{mkt}_{t,T,K} &= V^{mkt}_{t+\Delta t,T,K} - V^{mkt}_{t,T,K} \\ \Delta S_t &= S_{t+\Delta t} - S_t. \end{aligned} \tag{2.1.1}$$

The local hedging risk in the rebalancing interval Δt is:

$$\text{Risk}_{t,T,K}^{local} = \Delta S_t \delta_{t,T,K} - \Delta V_{t,T,K}^{mkt} \tag{2.1.2}$$

where $\delta_{t,T,K}$ is a hedging position for the corresponding option.

The discrete local hedging risk can be understood as the following. We set up the following portfolio at time t:

- A short position on option $V_{t,T,K}^{mkt}$
- A position of $\delta_{t,T,K}$, shares on underlying S_t,
- An amount in the risk-free bank.account B_t

We use $P_{t,T,K}^{H}$ to denote the hedging portfolio value at t for hedging options with expiry T and strike K. The hedging portfolio value is set to be zero by choosing B_t:

$$P_{t,T,K}^{H} = -V_{t,T,K}^{mkt} + S_t \delta_{t,T,K} + B_t = 0.$$

Therefore, the bank account at time t is set to be:

$$B_t = V_{t,T,K}^{mkt} - S_t \delta_{t,T,K}.$$

Let $dV_{t,T,K}^{mkt}$, dS_t, dB_t, and $dP_{t,T,K}^{H}$ denote the instantaneous change in the market option price, the underlying price, bank account and the hedging portfolio value respectively, the instantaneous hedging risk at time t is therefore:

$$dP_{t,T,K}^{H} = \ dS_t \delta_{t,T,K} - dV_{t,T,K}^{mkt} + dB_t. \tag{2.1.3}$$

Note that dB_t is deterministic when we assume r is a constant:

$$dB_t = rB_t dt$$

Therefore, if we assume the risk-free interest rate is zero (or omit dB_t since it is deterministic), we have:

$$dP_{t,T,K}^{H} = \ dS_t \delta_{t,T,K} - dV_{t,T,K}^{mkt}. \tag{2.1.4}$$

Under the assumption of a complete market where continuous hedging is feasible, the randomness in this instantaneous hedging risk can theoretically be eliminated by continuously trading the underlying asset and set $\delta_{t,T,K} = \frac{dV_{t,T,K}^{mkt}}{dS_t}$. However in practice, market is incomplete, since hedging can only be done at discrete times and additional risk, e.g., jump and volatility, cannot be eliminated by trading the underlying asset only [103, 84]. Lastly, in practice, $\frac{\partial V_{t,T,K}^{mkt}}{\partial S_t}$ is unobserved. Therefore, the hedging risk cannot be eliminated even instantaneously.

In practice, one actually care about the portfolio changes after a discrete time interval Δt. With the assumption of $r = 0$, after the fixed interval Δt, we have:

$$\Delta P^H_{t,T,K} = P^H_{t+\Delta t,T,K} - P^H_{t,T,K} = \Delta S_t \delta_{t,T,K} - \Delta V^{mkt}_{t,K,T}. \tag{2.1.5}$$

The local hedging risk is therefore defined to be the one-step hedging error when we assume the risk-free interest risk is zero. The local discrete hedging risk (2.1.2) measures the changes in the hedging portfolio after a fixed time interval Δt when the hedging position is set to be $\delta_{t,T,K}$. As $\Delta t \to 0$, we also have local hedging risk converge to instantaneous hedging risk.

2.1.2 Total Hedging Risk

In reality, one usually want to hedge until the expiries of the options, which requires re-balancing the hedging portfolio multiple times. Again, consider a hedging portfolio $P^H_{t,T,K}$ which is composed of:

- A short position on option $V^{mkt}_{t,T,K}$,
- A position of $\delta_{t,T,K}$ shares on underlying S_t,
- An amount in a risk-free bank account B_t.

For the notational simplicity since the T and K are fixed in following discussion, we drop them in the subscript:

$$V^{mkt}_t = V^{mkt}_{t,T,K}\,, \delta_t = \delta_{t,T,K},\ P^H_t = P^H_{t,T,K}.$$

Assume we rebalance N_{rb} times, the hedging portfolio is rebalanced at discrete times $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$ and the risk-free interest rate is $r = 0$. Initially at t_0, we have

$$P^H_{t_0} = -V^{mkt}_{t_0} + \delta_{t_0} S_{t_0} + B_{t_0} = 0$$

And

$$B_{t_0} = V^{mkt}_{t_0} - \delta_{t_0} S_{t_0}$$

At each rebalancing time t_j, we update our hedging position by changing the share we hold from $\delta_{t_{j-1}}$ to δ_{t_j} at t_j, where any required cash is borrowed, and any excess cash is loaned. Assume $\Delta t = t_j - t_{j-1}$ is fixed and risk-free interest rate is zero. The bank account is updated by:

$$B_{t_j} = B_{t_{j-1}} - S_{t_j}(\delta_{t_j} - \delta_{t_{j-1}})$$

Let t_j^+ and t_j^- be the time immediately after and immediately before t_j. Assume that the performance is measured at the $t_{N_{rb}}^-$, where $t_{N_{rb}} = T$ is the expiry:

$$\begin{aligned} P_{t_{N_{rb}}^-} &= B_{t_{N_{rb}-1}} - V_{t_{N_{rb}}}^{mkt} + S_{t_{N_{rb}}} \delta_{t_{N_{rb}-1}} \\ &= \sum_{j=0}^{N_{rb}-1} \left\{ \left(S_{t_{j+1}} - S_{t_j} \right) \delta_{t_j} \right\} + V_{t_0}^{mkt} - V_{t_{N_{rb}}}^{mkt} \\ &= \sum_{j=0}^{N_{rb}-1} \left\{ \left(S_{t_{j+1}} - S_{t_j} \right) \delta_{t_j} - (V_{t_{j+1}}^{mkt} - V_{t_j}^{mkt}) \right\} \end{aligned} \tag{2.1.6}$$

Equation (2.1.6) is defined as the *discrete total hedging risk*. If we always set $\delta_t = \frac{\partial V_t^{mkt}}{\partial S_t}$ and let $\Delta t \to 0$ (we continuously rebalance the portfolio), then $P_{t_{N_{rb}}^-} = 0$. In reality, even if we can set the hedging position to be $\delta_t = \frac{\partial V_t^{mkt}}{\partial S_t}$, we can only rebalance discretely due to the existence of transaction cost. Thus, $P_{t_{N_{rb}}^-}$ can take positive (profit) and negative value (loss). The *total hedging risk* measures the hedging portfolio profit and loss at the expiry T for the entire hedging period $[t_0, T]$.

Equation (2.1.6) can be written as:

$$\begin{aligned} P_{t_{N_{rb}}^-}^H &= \sum_{j=0}^{N_{rb}-1} \left\{ \left(S_{t_{j+1}} - S_{t_j} \right) \delta_{t_j} - (V_{t_{j+1}}^{mkt} - V_{t_j}^{mkt}) \right\} \\ &= \sum_{j=0}^{N_{rb}-1} \left\{ \Delta S_{t_j} \delta_{t_j} - \Delta V_{t_j}^{mkt} \right\} \end{aligned}$$

with $\Delta V_{t_j}^{mkt}$ and ΔS_{t_j} given in (2.1.1).

Plugging in the T and K into the subscript, we denote the discrete total hedging risk as:

$$\text{Risk}_{t_0,T,K}^{total} = \sum_{j=0}^{N_{rb}-1} \left\{ \Delta S_{t_j} \delta_{t_j,T,K} - \Delta V_{t_j,T,K}^{mkt} \right\} = \sum_{j=0}^{N_{rb}-1} \text{Risk}_{t_j,T,K}^{local} \tag{2.1.7}$$

which corresponding to the hedging portfolio value at the expiry T. By comparing equation (2.1.2) and (2.1.7), we can see that the *discrete total hedging risk* is the summation of the *discrete local hedging risk* evaluated at discrete rebalancing time $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$.

2.1.3 Interest Rate and Dividend

In the previous discussion in section 2.1.1 and section 2.1.2, we assume zero interest rate when we define the *discrete total hedging risk* and the *discrete local hedging risk*. The zero interest rate was assumed when measuring the hedging performance under discrete local hedging risk

framework. The results presented in [143] and [144] were evaluated from 2007 to 2015 for local hedging risk framework. During most of the time from 2007 and 2015, the interest rate is virtually zero. So the zero interest assumption is valid during most of the time. For the experimental results for total hedging risk in [145], such assumption is relaxed and we include the effect of interest rate since the period from 2000 to 2007 is also included in the experiments. We assume the interest rate is fixed as a constant during the life time of the hedging portfolio and denote it as r, the discrete total hedging risk is therefore:

$$\text{Risk}^{total}_{t_0,T,K} = \sum_{j=0}^{N_{rb}-1} \left\{ \left[\frac{S_{t_{j+1}}}{D(t_{j+1},T)} - \frac{S_{t_j}}{D(t_j,T)} \right] \delta^{M}_{t_j,T,K} \right\} + \frac{V^{mkt}_{t_0,T,K}}{D(t_0,T)} - V^{mkt}_{T,T,K} \tag{2.1.8}$$

where

$$D(t,T) = e^{-r(T-t)}$$

In this thesis, we focus on S&P 500 index options. For the S&P 500 index, the closing index price of each day is already adjusted to capture corporate actions that affect market capitalization such as as additional share insurance, dividends and restructuring events such as mergers or spin-offs[123]. When we calibrating the option pricing models such as the Black-Scholes model, an estimate of the dividends to be paid up until the expiration of the option is needed. We assume that the the index pays dividends continuously, according to a continuously-compounded annual dividend yield q. Similarly, traditional option pricing models requires a continuously-compounded interest rate as input. This interest rate is calculated from a collection of continuously compounded zero-coupon interest rates at various maturities, collectively referred to as the zero curve.

In this thesis, we use the option market data from OptionMetric [147] database. The OptionMetric [147] database provides us market option bid and ask quotes for each trading day from 1996-01 to 2015-08-31. We also use the zero curves on each trading day provided by the OptionMetric [147] database to extract the risk-free interest rate r. The zero curve used by the OptionMetric database is derived from ICE IBA USD LIBOR rates and settlement prices of CME Eurodollar futures. For a given option, the appropriate interest rate input r for option pricing corresponds to the zero-coupon rate that has a maturity equal to the time to maturity of the option $T-t$, and is obtained by linearly interpolating between the two closest zero-coupon rates on the zero curve. In addition, OptionMetric [147] database provides annual dividend yield q for the S&P 500 index. The q is recorded daily for the S&P 500 index and supplied as the input for calibrating the option pricing models. Details of how OptionMetric compute the zero curves and the annual dividend yield q using market data can be found in [147].

2.2 Option Pricing Model

2.2.1 Black-Scholes Model

In [20], the famous closed-form pricing Black–Scholes formula for European options is derived. Under Black–Scholes (BS) model, it is assumed that the underlying asset price follows a geometric Brownian motion:

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

where W_t is a standard Brownian motion, μ is the constant drift rate of the asset and σ is the constant volatility of the asset. We can easily show that:

$$S_t = S_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma W_t}$$

At any time t one can construct an instantaneously riskless portfolio consisting of one option and shares of the underlying asset. The riskless portfolio needs to be continuously adjusted so that the number of shares always equal to the partial derivative of the option pricing function with regards to the underlying asset. No-arbitrage condition implies that the the return of the riskless portfolio must be equal to the risk-free interest rate. This leads to the renowned Black-Scholes (BS) partial differential equation and the closed-form pricing formula.

Let C_{BS} be the option value function for call option which at time t and with underlying price S. For notational simplicity, we have:

$$\frac{\partial C_{BS}}{\partial t} = \frac{\partial C_{BS}}{\partial t}(t, S_t) \ , \frac{\partial C_{BS}}{\partial S_t} = \frac{\partial C_{BS}}{\partial S}(t, S_t) \ , \frac{\partial^2 C_{BS}}{\partial S_t^2} = \frac{\partial^2 C_{BS}}{\partial S^2}(t, S_t) \ , C_t^{BS} = C_{BS}(t, S_t)$$

With Ito's Lemma [165], we have:

$$dC_t^{BS} = \left(\frac{\partial C_{BS}}{\partial t} + \mu S_t \frac{\partial C_{BS}}{\partial S_t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 C_{BS}}{\partial S_t^2} \right) dt + \sigma S_t \frac{\partial C_{BS}}{\partial S_t} dW_t$$

Now consider a specific portfolio, called the delta-hedge portfolio, consisting of being short one call option and long $\frac{\partial C_{BS}}{\partial S_t}$ shares at time t. The total value of the delta-hedge portfolio P^δ at time t is:

$$P_t^\delta = -C_t^{BS} + S_t \frac{\partial C_{BS}}{\partial S_t}$$

The instantaneous profit or loss is:

$$dP_t^\delta = -\left(\frac{\partial C_{BS}}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 C_{BS}}{\partial S_t^2} \right) dt$$

Assume there is a riskless asset with risk-free rate of return r. We can see that the delta-hedge portfolio P_δ is also riskless because the diffusion term associated with dW_t is dropped. Under no-arbitrage condition, two riskless investment must earn the same rate of return so we must

have:

$$dP_t^{\delta} = rP_t^{\delta}dt$$

This leads to the Black-Scholes partial differential equation:

$$\frac{\partial C_{BS}}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 C_{BS}}{\partial S_t^2} + rS\frac{\partial C_{BS}}{\partial S_t} - rC_t^{BS} = 0 \tag{2.2.1}$$

The solution with European call option is the well-known Black-Scholes pricing formula:

$$C_{BS}(t,S) = S\mathcal{N}(d_1) - e^{-r(T-t)}K\mathcal{N}(d_2) \tag{2.2.2}$$

where $\mathcal{N}$ is the cumulative density function of the standard normal distribution

$$d_1 = \frac{\ln(S/K) + (r+\sigma^2/2)(T-t)}{\sigma\sqrt{T-t}},\ d_2 = d_1 - \sigma\sqrt{T-t}$$

Similarly, the Black-Scholes pricing formula for European put option is:

$$P_{BS}(t,S) = e^{-r(T-t)}K\mathcal{N}(-d_2) - S\mathcal{N}(-d_1) \tag{2.2.3}$$

Alternatively, we can derive the Black-Scholes formula under the risk-neutral pricing framework. As the name suggests, under a risk-neural measure Q, all agents in the economy are neutral to risk. Under a risk-neutral measure, all tradable assets have the same expected rate of return as the risk-free asset, which earns the risk-free interest rate r. The derivative price can thus derived from the expected payoff, discounted back to the current time at the risk-free rate r.

Under the Black-Scholes assumption, one can use the Girsanov's theorem [165] to convert the geometric brownian motion in the actual physical probability measure to the geometric brownian motion in a unique risk-neutral probability measure Q. Under risk-neutral pricing framework, we have:

$$C_{BS}(t,S) = e^{-r(T-t)}E^Q[\max(S_T - K, 0)] \tag{2.2.4}$$

$$P_{BS}(t,S) = e^{-r(T-t)}E^Q[\max(K - S_T, 0)] \tag{2.2.5}$$

where $E^Q[\cdot]$ is the expectation under the risk-neutral measure Q. More specifically, define Θ_1 to be the market price of risk:

$$\Theta_1 = \frac{\mu - r}{\sigma} \tag{2.2.6}$$

We change the original Brownian motion dW_t in the actual physical probability measure to $d\hat{W}_t$ in the risk-neutral probability measure Q with

$$d\hat{W}_t = dW_t + \Theta_1 dt \tag{2.2.7}$$

The underlying dynamic will have the drift to be the risk-free interest rate r.

$$dS_t = rS_t dt + \sigma S_t d\hat{W}_t$$

It can be shown that $d\hat{W}_t$ is the Brownian motion under the risk-neutral measure Q defined through Radon-Nikodym derivative via Girsanov's theorem [165]. Using the fact that the drift rate of underlying asset dynamic under risk-neutral measure Q is r, following (2.2.4) and (2.2.5), we can arrive at the same pricing formula as (2.2.2) and (2.2.3). Interest reader can refer to [165] for more details about risk-neutral pricing and change from physical measure to risk-neutral measure Q.

Since actual drift μ is irrelevant in determining the option price under Black-Scholes framework, when pricing the option using parametric models such as Black-Scholes model in this thesis, the drift of underlying dynamic is the risk-free interested r. Lastly, to include the effect of dividend, we can rewrite the d_1 and d_2 in the Black-Scholes pricing formula as:

$$d_1 = \frac{\ln(S/K) + ((r-q) + \sigma^2/2)(T-t)}{\sigma\sqrt{T-t}}, \;\; d_2 = d_1 - \sigma\sqrt{T-t}$$

where q is the given annual dividend yield. In this thesis, we use $V_{BS}(S,t,T,K,r,q;\sigma)$ to denote the European Black-Scholes pricing function regardless of the call or put nature.

Although, Black-Scholes framework provides a close-form formula, the Black-Scholes model is only a simple approximation to the market. Empirical evidence indicates markets often violate the assumption of Black-Scholes model. The two major aspects that has been criticized about Black-Scholes model are:

1. The constant volatility does not hold in real market. In practice, the implied volatility σ^{imp}, which equates the Black-Scholes option value $V_{BS}(S,t,T,K,r,q;\sigma)$ to market option price $V^{mkt}_{t,T,K}$, is often used to make sure that Black-Scholes price match the market observation. However, one can often find that the implied volatility σ^{imp} tends to differ across different strikes and expiries. This breaks down the assumption of a constant volatility

2. Transaction cost exists in a real market. Due to the existence of transaction cost, continuously adjusting the shares of underlying is not feasible and frequent hedging can be prohibitively expensive. Therefore, the argument of the Black-Scholes theory falls apart.

The violations from the assumption of Black-Scholes model in actual market motivate people to propose various approaches for relaxing the assumptions. These attempts include, but not limited to, local volatility models [45, 66, 158, 68], stochastic volatility models [99, 103, 111, 13], jump diffusion models [100, 122] and nonparametric pricing models based on regression [184, 17, 93, 82, 131]. Although nonparametric pricing models based on regression models can be a useful alternative, one usually ignores the no-arbitrage conditions in pricing when applying those regression approaches [184, 17, 93, 82, 131], which can be problematic in practice [115]. Recent focus of the data-driven models on pricing is on modelling and predicting the implied volatility

surface to ensure the absence of arbitrage [30, 192, 127]. In the following section, we discuss two stochastic volatility models relevant to this thesis, Heston model and SABR model, which provide efficient closed-form solutions for the option price similar to Black-Scholes model.

2.2.2 Heston Model

Heston [103] proposed a stochastic volatility model, which has often been used to model the volatility smile. The volatility smiles refer to the phenomenon that the options in real market whose strike prices differ substantially from the current underlying asset price tend to have higher implied volatilities than options whose strike prices are close to the underlying asset price. One of the key reasons for the popularity of the Heston model is that European call and put option under Heston model have closed-form solution which makes the calibration of the model computationally more efficient and more accurate. The Heston model assumes that the underlying, S_t follows a Black-Scholes type stochastic process, but with a stochastic variance Υ_t that follows a Cox, Ingersoll, Ross (CIR) process [55].

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{\Upsilon} S_t dW_t \\ d\Upsilon_t &= \kappa(\overline{\Upsilon} - \Upsilon_t)dt + \eta\sqrt{\Upsilon_t} dZ_t \\ E[dZ_t dW_t] &= \rho dt \end{aligned}$$

These parameters are described as follows:

- μ is the drift coefficient of the underlying asset
- $\overline{\Upsilon}$ is the long term mean of variance
- κ is the rate of mean reversion
- η is the volatility of volatility
- S_t is the underlying asset price
- Υ_t is the instantaneous variance
- W_t and Z_t are correlated Wiener processes with correlation coefficient ρ

Similarly, the Heston dynamics can be described under a risk-neutral measure Q. Heston [103] assumes that the market price of volatility risk is proportional to the volatility $\sqrt{\Upsilon_t}$:

$$\Theta_2 = \frac{\lambda}{\eta}\sqrt{\Upsilon_t} \tag{2.2.8}$$

λ is a parametric adjustment to the market price of volatility risk. Recall that market price of risk is:

$$\Theta_1 = \frac{\mu - r}{\sqrt{\Upsilon_t}}$$

It can be shown that a risk-neutral measure Q can be defined through Radon-Nikodym derivative via multi-dimensional Girsanov's theorem [165] using Θ_1 and Θ_2. Heston model under a risk-neutral measure Q is:

$$\begin{aligned} dS &= rSdt + \sqrt{\Upsilon} S d\hat{W} \\ d\Upsilon &= \kappa^*(\overline{\Upsilon}^* - \Upsilon)dt + \eta\sqrt{\Upsilon} d\hat{Z} \\ E[d\hat{Z}d\hat{W}] &= \rho dt \end{aligned} \tag{2.2.9}$$

where

$$\kappa^* = \kappa + \lambda, \overline{\Upsilon}^* = \frac{\kappa\overline{\Upsilon}}{\kappa+\lambda}$$
$$d\hat{W} = dW + \Theta_1 dt$$
$$d\hat{Z} = dZ + \Theta_2 dt$$

Similar to the Black-Scholes model, Heston model has the closed-form solutions. The closed formed solution for European call option is

$$C_{Heston}(t,S) = S\,N_1 - Ke^{-r(T-t)}\,N_2 \tag{2.2.10}$$

Let us define the imaginary unit $\mathscr{I}^2 = -1$. Then the N_1 and N_2 are defined as:

$$N_j = \frac{1}{2} + \frac{1}{\pi}\int_0^\infty Re\left[\frac{e^{-\mathscr{I}\varphi \ln K} f_j(S,\Upsilon,t,T;\varphi)}{\mathscr{I}\varphi}\right] d\varphi; \;\; j = 1,2$$

with $Re[\cdot]$ denoting the real part of a complext number. The characteristic function $f_j(S,\Upsilon,t,T;\varphi)$ is :

$$f_j(S,\Upsilon,t,T;\varphi) = e^{A_j(t,T,\varphi) + B_j(t,T,\varphi)\Upsilon + \mathscr{I}\varphi \ln S}, j = 1,2$$

where:

$$A_j(t,T,\varphi) = r\varphi(T-t) + \frac{\kappa^*\overline{\Upsilon}^*}{\eta^2}\left\{(b_j - \rho\eta\varphi\mathscr{I} + d_j)(T-t) - 2\ln\left[\frac{1-g_j e_j^d(T-t)}{1-g_j}\right]\right\}$$
$$B_j(t,T,\varphi) = g_j\left[\frac{1-e_j^d(T-t)}{1-g_j e^{d_j(T-t)}}\right]$$
$$g_j = \frac{b_j - \rho\varphi\mathscr{I} + d_j}{b_j - \rho\varphi\mathscr{I} - d_j}$$
$$d_j = \sqrt{(\rho\eta\varphi\mathscr{I} - b_j)^2 - \eta^2(2u_j\varphi - \varphi^2)}$$
$$u_1 = 0.5, u_2 = -0.5, b_1 = \kappa^* - \rho\Upsilon, b_2 = \kappa^*$$

European put option price can be derived from the call-put parity:

$$P_{Heston}(t,S) = C_{Heston}(t,S) - S + Ke^{-r(T-t)}$$

The parameters to be calibrated from the market option prices are $\{\Upsilon_0, \kappa^*, \overline{\Upsilon}^*, \eta, \rho\}$ where Υ_0 is the initial instantaneous variance, $\overline{\Upsilon}^*$ is the long term mean of variance under risk-neutral measurem Q, κ^* is the rate of mean reversion under risk-neutral measure Q, η is the volatility of volatility, and ρ is correlation.

Under the assumption of the Black-Scholes model, an option is written on a tradable asset S_t. The randomness in option value is determined by the randomness of the asset S_t. Such uncertainly can be hedged by continuously adjusting the shares of underlying asset as we have discussed in section 2.2.1. This implies a complete market [165]. Under a stochastic volatility model such as Heston model, the uncertainty of option value comes from both the underlying asset S_t and the volatility (or the variance Υ_t as in Heston model). The volatility itself is not tradable which implies an incomplete market under stochastic volatility model. One can assume a risk-neutral measure Q exists and calibrate the Heston model to match the the market option prices directly using the dynamics in (2.2.9) without specifying λ. In this way, λ has been implied and embedded into the calibrated model parameters κ^* and $\overline{\Upsilon}^*$. Interested readers can refer to [103, 84] for more details of risk-neutral pricing under the Heston model.

In this thesis, we deal with Heston model under the risk-neural measurement Q. For simplicity, in this thesis, we use $V_{Heston}(S,t,T,K,r,q;\Upsilon_0,\kappa^*,\overline{\Upsilon}^*,\eta,\rho)$ to denote the European Heston pricing function regardless whether it is a call option or a put option.

2.2.3 SABR Model

The SABR model [99] is another stochastic volatility model, which attempts to capture the volatility smile in a derivative market. The name SABR stands for "Stochastic Alpha, Beta, Rho", referring to the parameters of the model. The popularity of SABR model is due to the fact that it can reproduce the market-observed volatility smile more accurately. Additionally, it provides a closed-form formula for the implied volatility under the Black model, which is a variant of the Black–Scholes model. Given the risk-free interest rate r, the annual dividend yield q, and the forward F_t with expiry T is:

$$F_t = S_t e^{(r-q)(T-t)}$$

In the SABR stochastic volatility model, the forward F_t price follows the following stochastic differential equation:

$$\begin{aligned} dF_t &= \alpha_t (F_t)^\beta dW_t \\ d\alpha_t &= \nu \alpha_t dZ_t \\ E[dW_t dZ_t] &= \rho dt \end{aligned}$$

These parameters are described as follows:

- α_t is the instantaneous volatility of the forward F_t.
- ν is the volatility of instantaneous volatility α_t .
- W_t and Z_t are correlated Wiener processes with correlation coefficient ρ

A variant of the Black–Scholes option pricing model, the Black model [19], is often used together with SABR model. Under thr Black model, the forward F_t price follows the following stochastic differential equation:

$$dF_t = \sigma_B F_t dW_t$$

where σ_B is the volatility. We use $V_B(F,K,r,t,T;\sigma_B)$ to denote the Black pricing function. For European call option:

$$C_B(F,t,T,K,r;\sigma_B) = e^{-r(T-t)}[F\mathcal{N}(d_3) - K\mathcal{N}(d_4)]$$

For European put option:

$$P_B(F,t,T,K,r;\sigma_B) = e^{-r(T-t)}[K\mathcal{N}(-d_4) - F\mathcal{N}(-d_3)]$$

where $\mathcal{N}$ is the cumulative density function of standard normal distribution

$$d_3 = \frac{\ln(F/K) + \sigma^2/2(T-t)}{\sigma\sqrt{T-t}},\ d_4 = d_3 - \sigma\sqrt{T-t}$$

Consider an option on the forward F with expiry T and strike K at time t. If we force the SABR model price of the option into the form of the Black model valuation formula, then the SABR implied volatility, which is the value of the σ_B in Black's model that forces it to match the SABR price, is approximately given by [99, 16]:

$$\begin{aligned}\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho) \approx & \frac{\alpha}{(FK)^{(1-\beta)/2}\left[1 + \frac{(1-\beta)^2}{24}\log^2(F/K) + \frac{(1-\beta)^4}{1920}\log^4(F/K) + \cdots\right]} \cdot \frac{z}{x(z)} \\ & \cdot \left\{1 + \left[\frac{(1-\beta)^2}{24}\frac{\alpha^2}{(FK)^{1-\beta}} + \frac{1}{4}\frac{\rho\beta\nu\alpha}{(FK)^{(1-\beta)/2}} + \frac{2-3\rho^2}{24}\nu^2\right](T-t)\right\}\end{aligned} \tag{2.2.11}$$

where

$$z = \frac{\nu}{\alpha}(FK)^{(1-\beta)/2}\log(F/K),\ x(z) = \log\left\{\frac{\sqrt{1-2\rho z+z^2}+z-\rho}{1-\rho}\right\}$$

For the special case of at-the-money options, which are the options struck at $K = F$, this formula

reduces to

$$\begin{aligned}\sigma_{ATM} &= \sigma_B(F,t,T,F;\alpha,\beta,\nu,\rho) \\ &\approx \frac{\alpha}{F^{(1-\beta)}}\left\{1+\left[\frac{(1-\beta)^2}{24}\frac{\alpha^2}{F^{2-2\beta}}+\frac{1}{4}\frac{\rho\beta\alpha\nu}{F^{(1-\beta)}}+\frac{2-3\rho^2}{24}\nu^2\right](T-t)\right\}\end{aligned}$$

Therefore, the European option value under SABR model is given by:

$$C_{SABR} = C_B(F,t,T,K;\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho))$$

$$P_{SABR} = P_B(F,t,T,K;\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho))$$

The $\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho)$ under SABR model depends on the forward F, the strike K, the time to expiry $T-t$, the initial SABR volatility α, the power of forward β, the volatility of volatility ν, and the correlation ρ. Interested readers can refer to [99] for the detailed derivation of the analytical approximation $\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho)$. [1] We use $V_{SABR}(S,t,T,K,r,q;\alpha,\beta,\nu,\rho)$ to denote the option pricing formula under SABR model regardless whether it is a call toption and a put option in the latter discussion.

2.3 Option Hedging Using the Sensitivity from Pricing Model

Ideally, we could compute the $\frac{\partial V^{mkt}}{\partial S}$ as the instantaneous hedging position. In reality, $\frac{\partial V^{mkt}}{\partial S}$ is unknown. Thus we often assume a specific underlying dynamics as what has been discussed in 2.2. In order to obtain the pricing model parameters, we calibrate the model so that model option prices can match the market option prices. Then one can use the $\frac{\partial V}{\partial S}$ as the hedging position. This is know as the delta-hedging strategy.

In this section, we discuss how to compute the hedging position $\delta_{t,T,K}$ from the option pricing model under the delta-hedging strategy, as well as the drawbacks of this approach. Specifically, we discuss the pricing model parameters dependence issue, which motivates our data-driven model. In addition, we provide several existing ways for correcting the model parameter dependence.

2.3.1 Classical Hedging Position from Pricing Model

In section 2.2, we have introduced several option pricing models with analytical pricing formula. In this section, we consider the option hedging. We assume the risk-free interest rate and annual dividend yield are given as r and q. Therefore, we denote $V(S,t,T,K;\theta)$ as an option pricing model with θ as the model parameters to be calibrated. For Black-Scholes model, θ is the

[1] Note that equation (2.2.11) is an approximation where higher order terms are dropped, we will disucss in later chapter on how to fix the potential arbitrage created by the approximation.

volatility σ. For Heston model, $\theta = \{\Upsilon_0, \kappa^*, \overline{\Upsilon}^*, \eta, \rho\}$. For SABR model, $\theta = \{\alpha, \beta, \nu, \rho\}$. [2] In order to determine hedging position with the most recent information, we calibrate the pricing model to match the market option prices.

On each trading time t, we have observed market option prices for different strikes $\{K_1, \ldots, K_{N_k(t)}\}$ and different expiries $\{T_1, \ldots, T_{N_T(t)}\}$, where $N_k(t)$ is the number of strikes and $N_T(t)$ is the number of expiries. Note that we use $N_k(t)$ and $N_T(t)$ to indicate the number of strikes and number of expiries observed from market are time-dependent. The underlying asset price is S_t. The calibration can be done by:

$$\min_{\theta} \sum_{i=1}^{N_T(t)} \sum_{j=1}^{N_K(t)} \left(V(S_t, t, T_i, K_j; \theta) - V^{mkt}_{t,T_i,K_i} \right)^2 \tag{2.3.1}$$

Determining a single model solving equation (2.3.1) is chanllenging. In practice. approximations are often made. For the Black-Scholes model, there is only one parameter to be calibrated, so one can match each market prices exactly by:

$$V_{BS}(S_t, t, T_i, K_j; \sigma^{imp}_{t,T_i,K_j}) = V^{mkt}_{t,T_i,K_i}$$

The volatility σ^{imp}_{t,T_i,K_j} that equals the Black–Scholes price with the market option price is called **Black-Scholes implied volatility**. (In section 2.2.3, we have introduced a term called SABR implied volatility which equals the SABR model price with the Black model price. The SABR implied volatility and Black-Scholes implied volatility are two different but related concepts.) Note that in this way, for different options V^{mkt}_{t,T_i,K_j}, different models are calibrated. One can often find that the implied volatility σ^{imp} tends to differ across different strikes and expiries.

For stochastic volatility models, such as Heston and SABR, the calibration is usually done by matching the volatility smile for each expiry T_i instead of the entire volatility surface [103, 99, 112]. In other words, for a expiry T_i, we solve

$$\min_{\theta} \sum_{j=1}^{N_K(t)} \left(V(S_t, t, T_i, K_j; \theta) - V^{mkt}_{t,T_i,K_j} \right)^2$$

Note that in this way, pricing models are calibrated separately for different expiries T_i on the same trading date t.

Given the option pricing model $V(S, t, T, K; \theta^*)$ with the calibrated θ^*, the hedging position is commonly determined by:

$$\delta_{t,T,K} = \frac{\partial V(S, t, T, K; \theta^*)}{\partial S}$$

Although a more complex option pricing model such as stochastic volatility models can match the market prices better and account for the volatility smile better, it does not necessarily lead

[2] Note that, Studies [112, 99, 98] shows that β can often be fixed a priori. In this case, $\theta = \{\alpha, \nu, \rho\}$.

to better hedging strategy. Several studies [66, 13, 190, 123] detail the hedging performances of option pricing models and show that, while explicitly modelling volatility smile substantially improves the pricing accuracy, it frequently decreases the hedging performance, compared to the simpler Black-Scholes models. A recent empirical study [123] demonstrates that, for hedging S&P 500 index options, the simple Black-Scholes model with the implied volatility calibrated on each trading date can even outperform complex stochastic volatility models such as Heston model [103] and Heston-Nandi model [104]. The phenomenon where better pricing model leads to worse hedging performance is refer to as pricing/hedging conundrum in [123]. In the following section, we provide one of the potential explanation why such phenomenon exists.

2.3.2 Parameter Dependence on Underlying Asset Price

We demonstrate that determining the hedging position from the partial derivative:

$$\delta_{t,T,K} = \frac{\partial V(S,t,T,K;\theta^*)}{\partial S}$$

of the calibrated pricing function inevitably leads to incomplete elimination of the underlying sensitivity, even in the context of instantaneous hedging. To see this, let us assume that a pricing model matches the market price $V^{mkt}_{t,T,K}$ with strike K and expiry T at t exactly, i.e., θ^* is determined so that:

$$V(S,t,T,K;\theta^*) = V^{mkt}_{t,T,K}. \tag{2.3.2}$$

Let us further assume the risk-free interest rate and dividend yield $\{r,q\}$ do not depend on S.

The derivative pricing theory yields the risk neutral (or risk-adjusted) option value function $V(\cdot)$. The option hedging position is then often determined by the model option function sensitivity $\frac{\partial V}{\partial S}$. However, using $\frac{\partial V}{\partial S}$ as the hedging position only ensures delta neutral with respect to the model option function V. Unfortunately this does not ensure delta neutral with respect to the market price V^{mkt} . This is because that the calibration equation (2.3.2) does not imply that the sensitivity $\frac{\partial V}{\partial S}$ matches $\frac{\partial V^{mkt}}{\partial S}$, since the calibration equation contains no information on change in the market option price. This leads to dependence of the parameters of the calibrated model on the underlying price, i.e., $\frac{\partial \theta^*}{\partial S} \neq 0$. This can be understood by hypothetically assuming that (2.3.2) holds at any S, which implies

$$\frac{\partial V}{\partial S} + \frac{\partial V}{\partial \theta^*}\frac{\partial \theta^*}{\partial S} = \frac{\partial V^{mkt}}{\partial S} \tag{2.3.3}$$

Since calibration (2.3.2) only ensures matching in the option values, not matching the change in the market option price, it is likely that $\frac{\partial V}{\partial S} - \frac{\partial V^{mkt}}{\partial S} \neq 0$. Although this parameter dependence issue has been noted, see, e.g., [45, 112], correcting for this risk exposure is not straightforward.

Even with the perfect case where we can have a pricing model matching all observed market prices, which is often not achievable due to the calibration error following the equation (2.3.3), we still have the pricing model parameters dependence issue to be addressed. Since it is difficult

to account for parameter dependence $\frac{\partial \theta^*}{\partial S}$, $\frac{\partial V^{mkt}}{\partial S}$ can be hard to be captured accurately. Therefore, it is difficult for the classical delta-hedging approach of using $\frac{\partial V}{\partial S}$ as the hedging position to eliminate the instantaneous hedging risk.

A more complex model usually has more pricing model parameters to be calibrated. Therefore, the parameter dependence likely becomes harder to be addressed. That may be one of the reason why many studies observe poorer hedging performance from more complex pricing model than simple Black-Scholes model in the real market. Even with the simple Black-Scholes model, the Black–Scholes implied volatility still depends on the underlying asset. In the following section, we introduce several corrections under the Black-Scholes framework for the parameter dependence. In addition, complex models like Heston model usually can not match the observed market option prices exactly. The calibration error can also potentially increase the hedging errors when the partial derivative is used as the hedging position.

2.3.3 Corrections under the Black-Scholes Framework

Although the issue of pricing model parameters dependence on underlying asset has been noted, e.g., in [45, 112], correcting for this risk exposure is not straightforward. Methods have been proposed to compensate for this missing exposure. Typically methods are based on analysis of some parametric models, e.g., LVF and SV; these correction methods attempt to minimize the variance of the hedging error [112, 3, 7, 8, 92]. In this section, we present several correction methods under Black-Scholes framework. The discussion below is drawn from [112, 98, 16].

2.3.3.1 A Data-Driven Correction for the Black–Scholes Delta Hedging

Equation (2.3.3) implies that the sensitivity of the option value function $\frac{\partial V}{\partial S}$, calibrated at the rebalancing time to satisfy the calibration equation (2.3.2), cannot completely hedge the sensitivity of the market option price to the underlying price since $\frac{\partial \theta^*}{\partial S}$ is not accounted for. If the hedging risk in each rebalancing period is the performance criteria, the information in the option market price change needs to be explicitly incorporated to determine a better hedging method which minimizes the hedging risk as measured by the market option price changes instead of the model option price changes.

If sufficient market option and underlying price data exist, a potentially more effective approach is to learn the hedging strategy based on historical observations of both changes in the market option prices and the underlying prices. This approach is completely different from the existing sensitivity approach based on parametric option pricing model.

To the best of our knowledge, the study [112] is the first to determine the hedging strategy based on historical observations of the market option prices and underlying prices.

Hull and White [112] proposed a correction form based on the following analysis using the Black–Scholes model. More specifically, assume we have calibrated the implied volatilities σ^{imp} to match all the market option prices as what we have discussed in section 2.3.1. Let us again

assume we are computing the hedging position with a fixed T and K at a specific trading time t, so for notational simplicity in the following discussion, we drop t, T, K in the subscript. Following (2.3.3), to determine the sensitivity accurately, one needs to compute

$$\frac{\partial V_{\mathrm{BS}}}{\partial S} + \frac{\partial V_{\mathrm{BS}}}{\partial \sigma^{imp}} \frac{\partial \sigma^{imp}}{\partial S} \tag{2.3.4}$$

$\frac{\partial V_{\mathrm{BS}}}{\partial S}$ is the Black–Scholes delta with the implied volatility and $\frac{\partial V_{\mathrm{BS}}}{\partial \sigma^{imp}}$ is Black–Scholes vega with the implied volatility. Both can be computed analytically. The undetermined part is $\frac{\partial \sigma^{imp}}{\partial S}$. In [112], the authors choose

$$\frac{\partial \sigma^{imp}}{\partial S} = \frac{a + b \cdot \frac{\partial V_{\mathrm{BS}}}{\partial S} + c \cdot (\frac{\partial V_{\mathrm{BS}}}{\partial S})^2}{S\sqrt{T-t}}$$

where a, b and c are the parameters to be fitted using market option data. Therefore, plugging in the notation for t, T, K, the minimum variance delta is therefore:

$$\delta^{MV}_{t,T,K} = \delta^{BS}_{t,T,K} + \mathrm{vega}^{BS}_{t,T,K} \frac{a + b \cdot \delta^{BS}_{t,T,K} + c \cdot (\delta^{BS}_{t,T,K})^2}{S_t\sqrt{T-t}}, \tag{2.3.5}$$

where $\delta^{BS}_{t,T,K}$ and $\mathrm{vega}^{BS}_{t,T,K}$ are respectively the Black–Scholes delta and vega, using the implied volatility at time t for option with strike K and expiry T.

The model parameters a, b, c are determined using historical observations. Let M denote the number of historical data instances. Each data instance corresponds to a unique combination of $\{t, T, K\}$. For data instance i, with time t_i, strike K_i, and expiry T_i, we approximate the instantaneous changes in the market option price and underlying price by the **daily** changes of the market option prices and underlying prices:

$$dV^{mkt}_{t_i,T_i,K_i} \approx \Delta V^{mkt}_{t_i,T_i,K_i}, \;\; dS_{t_i} \approx \Delta S_{t_i}$$

Thus, we have the following minimization problem:

$$\min_{a,b,c} \sum_{i=1}^{M} \left(\Delta S_{t_i} \delta^{MV}_{t_i,T_i,K_i} - \Delta V^{mkt}_{t_i,T_i,K_i} \right)^2$$

A linear regression is performed to determine the model parameters a, b, c.

Consequently, the minimum variance hedging function δ^{MV} is computed based on regression estimation, assuming the quadratic parametric model (2.3.5). It has been shown [112] that the corrective formula (2.3.5) can significantly improve the daily hedging performance.

2.3.3.2 Correction With LVF Model For The Black–Scholes Delta Hedging

As an improvement over the BS model, the local volatility function (LVF) model,

$$\frac{dS_t}{S_t} = (r-q)dt + \sigma(S_t,t)dW_t \tag{2.3.6}$$

has also been considered, see e.g., [67, 60, 61]. LVF and its extensions remain widely popular in practice [46, 57]. Many methods have been proposed to calibrate a local volatility function $\sigma(S,t)$ from the traded market option prices, see e.g., [116, 4, 44]. In addition, Coleman et al. [45] discuss a relationship between the partial derivatives of calls and puts in the context of the LVF model, under which a call and put symmetry relation holds, see, e.g., [31, 32]. This relationship is found useful in correcting the dependence of the implied volatility on the underlying for Black–Scholes delta hedging.

In Theorem 2.3.1 below, we formalize this relation [45] to any call and put functions satisfying the call-put-symmetry. Let $Call(S,t,T,K,r,q)$ and $Put(S,t,T,K,r,q)$ be the call option price and the put option price with underlying price S, strike K, expiry T, time t, interest rate r and dividend yield q. Then let $Put(K,t,T,S,q,r)$ denote the put price with underlying price K, strike S, expiry T, time t , risk free rate q and dividend yield r. When the underlying price satisfies the stochastic equation (2.3.6), the European put and call prices are related through the reversal of K and S, and q and r via the following call-put symmetry.

$$Call(S,t,T,K,r,q) = Put(K,t,T,S,q,r). \tag{2.3.7}$$

Interested reader can refer to [95, 45] for the detailed proof of this call-put symmetry. We note that, for this relationship to be useful in correcting for MV hedging in practice, the relevant price functions correspond to the market option prices, not model option values.

Theorem 2.3.1. *Let $Call(S,t,T,K,r,q)$ and $Put(S,t,T,K,r,q)$ be the call option price and put option price with underlying price S, strike K, expiry T, time t, interest rate r and dividend yield q. Assume further that there exists a unique implied volatility calibrating to $Call(S,t,T,K,r,q)$ and $Put(S,t,T,K,r,q)$ respectively and the call-put-symmetry below holds:*

$$Call(S,t,T,K,r,q) = Put(K,t,T,S,q,r). \tag{2.3.8}$$

Then

$$\frac{\partial \breve{\sigma}_c(S,t,T,K,r,q)}{\partial S} = \frac{\partial \breve{\sigma}_p(K,t,T,S,q,r)}{\partial S}. \tag{2.3.9}$$

where $\breve{\sigma}_c(\cdot)$ is the BS implied volatility calibrated to $C(\cdot)$ and $\breve{\sigma}_p(\cdot)$ is the BS implied volatility calibrated to $P(\cdot)$ respectively.

Proof. Let $C_{\mathrm{BS}}(\cdot)$ and $P_{\mathrm{BS}}(\cdot)$ denote the BS model option value functions. Put and call symmetry under the Black-Scholes model implies that

$$C_{\mathrm{BS}}(S,t,T,K,r,q;\sigma) = P_{\mathrm{BS}}(K,t,T,S,q,r;\sigma), \tag{2.3.10}$$

where σ is any constant volatility. Let $\breve{\sigma}_c(S,t,T,K,r,q)$ and $\breve{\sigma}_p(K,t,T,S,q,r)$ be the Black–Scholes implied volatilities calibrated to $Call(S,t,T,K,r,q)$ and $Put(K,t,T,S,q,r)$. Then

$$\begin{aligned} Call(S,t,T,K,r,q) &= C_{\mathrm{BS}}(S,t,T,K,r,q;\breve{\sigma}_c(S,t,T,K,r,q)) \\ Put(K,t,T,S,q,r) &= P_{\mathrm{BS}}(K,t,T,S,q,r;\breve{\sigma}_p(K,t,T,S,q,r)). \end{aligned} \tag{2.3.11}$$

From (2.3.8) and above, it follows

$$C_{\mathrm{BS}}(S,t,T,K,r,q;\breve{\sigma}_c(S,t,T,K,r,q)) = P_{\mathrm{BS}}(K,t,T,S,q,r;\breve{\sigma}_p(K,t,T,S,q,r)) \tag{2.3.12}$$

Using (2.3.10) with $\breve{\sigma}_c(\cdot)$,

$$C_{\mathrm{BS}}(S,t,T,K,r,q;\breve{\sigma}_c(S,t,T,K,r,q)) = P_{\mathrm{BS}}(K,t,S,T,q,r;\breve{\sigma}_c(S,t,T,K,r,q))$$

From above and (2.3.12), it follows

$$P_{\mathrm{BS}}(K,S,T,q,r,\breve{\sigma}_c(S,t,T,K,r,q)) = P_{\mathrm{BS}}(K,t,T,S,q,r;\breve{\sigma}_p(K,t,T,S,q,r))$$

Assuming that there is a unique implied volatility from the BS formula, we have

$$\breve{\sigma}_c(S,t,T,K,r,q) = \breve{\sigma}_p(K,t,T,S,q,r) \tag{2.3.13}$$

Taking derivative with respect to S, we have

$$\frac{\partial \breve{\sigma}_c(S,t,T,K,r,q)}{\partial S} = \frac{\partial \breve{\sigma}_p(K,t,T,S,q,r)}{\partial S},$$

i.e., (2.3.9) holds. This completes the proof. □

Assume that the market option prices satisfy put-call symmetry. Then the relevance of Theorem 2.3.1 in accounting for dependence of the implied volatility on the underlying can be appreciated as follows. On the left hand side of (2.3.9), the derivative is with respect to the underlying price (the first argument). On the right hand side, the derivative is with respect to the strike price (the fourth argument). When $q \approx r$ (as in the futures options market), $\frac{\partial \breve{\sigma}_p(K,t,T,S,q,r)}{\partial S}$ can be estimated from the observed implied volatility surface. In other words, for at-the-money with $K = S$ option, the rate of change in the implied volatility with respect to changes in the underlying price is equal to the slope of the volatility smile. Consequently the sensitivity of the implied volatility to the underlying can be estimated.

Hull and White [112] implement a corrective formula for minimum variance hedging based on (2.3.9), referred to as the LVF minimum variance hedging. Hull and White [112] further assume that the rate of change in the implied volatility with respect to changes in the underlying price is equal to the slope of the volatility smile for options which are not at-the-money. Therefore, the LVF correction is:

$$\delta^{LVF}_{t,T,K} = \delta^{BS}_{t,T,K} + \text{vega}^{BS}_{t,T,K} \frac{\partial \sigma^{imp}}{\partial K} \tag{2.3.14}$$

where $\frac{\partial \sigma^{imp}}{\partial K}$ can be estimated from the a quadratic function fitting the volatility smile for each expiry. Other minimum variance delta hedge methods have also been proposed to correct practitioner Black–Scholes delta explicitly, see, e.g., [13, 56, 15, 152, 11]. Interested reader can refer to them for more details.

2.3.3.3 Correction With SABR model for the the Black–Scholes Delta Hedging

Recall that the SABR implied volatility, which is the value of the σ_B in Black's model that forces the Black's model price to match the SABR model price, is given in section 2.2.3. Based on SARR model, a correction formula for Black delta hedging is thus given by:

$$\frac{\partial V_B}{\partial F} + \frac{\partial V_B}{\partial \sigma_B} \frac{\partial \sigma_B}{\partial F} \tag{2.3.15}$$

Note that, Black delta can be converted to Black–Scholes delta. One can rewrite the formula (2.3.15) to compute the sensitivity with regards to underlying asset price S instead of forward F by:

$$\left(\frac{\partial V_B}{\partial F} + \frac{\partial V_B}{\partial \sigma_B} \frac{\partial \sigma_B}{\partial F} \right) \frac{\partial F}{\partial S} \tag{2.3.16}$$

Given the risk-free interest rate r, the annual dividend yield q, the forward F_t at time t with expiry T is:

$$F_t = S_t e^{(r-q)(T-t)}$$

Adding the dependence for t, T, K as what we did in previous sections, we have:

$$\delta^{SABR}_{t,T,K} = \left(\delta^{B}_{t,T,K} + \text{vega}^{B}_{t,T,K} \frac{\partial \sigma_B}{\partial F} \right) e^{(r-q)(T-t)} \tag{2.3.17}$$

where $\delta^{B}_{t,T,K}$ is the Black delta $\frac{\partial V_B}{\partial F}$ at time t for option with strike K and expiry T. Here $\text{vega}^{B}_{t,T,K}$ is the Black vega $\frac{\partial V_B}{\partial \sigma_B}$ at time t for the option with strike K and expiry T, and $\frac{\partial \sigma_B}{\partial F}$ is the shortened form of $\frac{\partial \sigma_B(F_t,t,T,K;\alpha,\beta,\nu,\rho)}{\partial F}$.

Although, SABR model can be used to correct for the dependence of implied volatility on the forward price, the formula (2.3.17) omits the fact the initial SABR volatility α is correlated with the forward F. Here whenever the forward F changes, the initial SABR volatility α changes. There is still unaccounted parameter dependence in equation (2.3.17).

Bartlett correction [98, 16] was proposed to account for the dependence of α on F. The Bartlett correction is based on the following analysis. Recall that, in the SABR model, we

assume:

$$\begin{aligned} dF_t &= \alpha_t (F_t)^\beta dW_t \\ d\alpha_t &= \nu \alpha_t dZ_t \\ E[dW_t dZ_t] &= \rho dt \end{aligned}$$

which can be rewritten as:

$$\begin{aligned} dF_t &= \alpha_t (F_t)^\beta dW_t \\ d\alpha_t &= \nu \alpha_t \left(\rho dW_t + \sqrt{1-\rho^2} d\hat{Z}_t \right) = \frac{\rho \nu}{(F_t)^\beta} dF_t + \nu \alpha_t \sqrt{1-\rho^2} d\hat{Z}_t \end{aligned}$$

where $\hat{Z}_t$ and W_t are independent random variables. Therefore, one can readily find that:

$$\frac{d\alpha_t}{dF_t} = \frac{\rho \nu}{(F_t)^\beta} + \frac{\nu \sqrt{1-\rho^2}}{(F_t)^\beta} \frac{d\hat{Z}_t}{dW_t}.$$

Since $\hat{Z}_t$ and W_t are independent, we have [16]:

$$E\left[\frac{d\alpha_t}{dF_t} \right] = \frac{\rho \nu}{(F_t)^\beta}$$

The formula for Bartlett correction is thus:

$$\delta^{Bartlett}_{t,T,K} = \left[\delta^{B}_{t,T,K} + \text{vega}^{B}_{t,T,K} \left(\frac{\partial \sigma_B}{\partial F} + \frac{\partial \sigma_B}{\partial \alpha} \frac{\rho \nu}{(F_t)^\beta} \right) \right] e^{(r-q)(T-t)} \tag{2.3.18}$$

with $\frac{\partial \sigma_B}{\partial \alpha}$ being the shortened form for $\frac{\partial \sigma_B(F_t,t,T,K;\alpha,\beta,\nu,\rho)}{\partial \alpha}$. The Bartlett correction improves over the SABR delta in two different aspects:

- Empirically, the Bartlett correction is less sensitive to the choice of β parameter. In practice, β is often fixed instead of being calibrated together with α, ν, ρ in SABR model. Different choices of β can often all fit the market prices but different choices of β can often lead to different SABR delta $\delta^{SABR}_{t,T,K}$ position. On the other hand, $\delta^{Bartlett}_{t,T,K}$ is consistent with different choices of β. This phenomenon is shown in Figure 2.1. We calibrate the SABR model for S&P 500 index options on 2012-01-04 and the option expiry is 2012-12-31. We compute and compare delta position and we can see that for SABR delta, different choice of β can lead to significantly different hedging position while the Bartlett delta position is consistent.

- Empirically, the Bartlett correction provides better hedging strategy. As an example, one can observe significant hedging performance difference between $\delta^{Bartlett}_{t,T,K}$ and $\delta^{SABR}_{t,T,K}$ on S&P 500 index options. More specifically, we calibrate the SABR models on each trading date from 2007-01-01 to 2015-08-31 as discussed in section 2.2 with $\beta = 1$ and note that

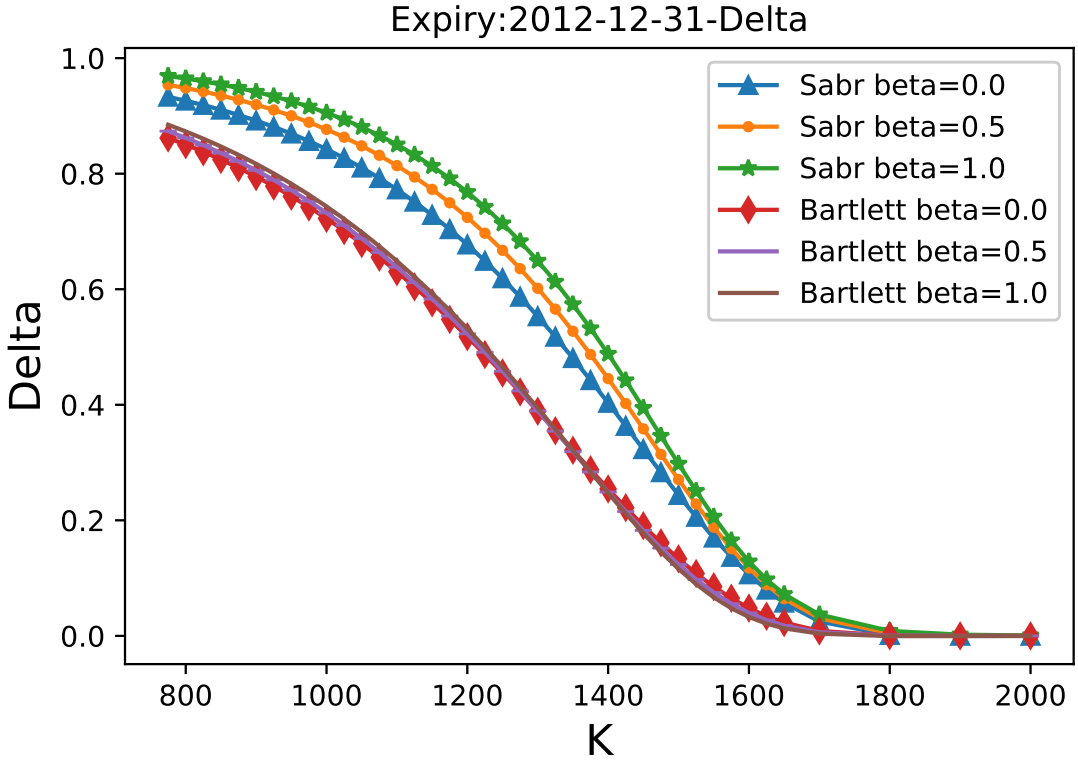


Figure 2.1: Comparing the original SABR delta $\delta_{t,T,K}^{SABR}$ and Bartlett delta $\delta_{t,T,K}^{Bartlett}$ from SABR models calibrated with $\beta = 0,\ 0.5,\ 1.0$ on 2012-01-04 for S&P500 index option. The computation of Bartlett delta $\delta_{t,T,K}^{Bartlett}$ is more robust to the choice of β.

here we calibrate SABR models separately for each expiry on each trading date t. And we compare the local hedging performance for all the traded options. Following [112], we use the Gain ratio below as the measure of the local hedging risk performance:

$$\text{GAIN} = 1 - \frac{\sum_{i=1}^{m} \left(\Delta V^{mkt}_{t_i,T_i,K_i} - \delta_{t_i,T_i,K_i} \Delta S_{t_i} \right)^2}{\sum_{i=1}^{m} \left(\Delta V^{mkt}_{t_i,T_i,K_i} - \delta^{BS}_{t_i,T_i,K_i} \Delta S_{t_i} \right)^2}. \tag{2.3.19}$$

where m is the total number of data instances to be evaluated. The $\delta^{BS}_{t_i,T_i,K_i}$ is the implied Black–Scholes delta, δ_{t_i,T_i,K_i} is the hedging position from the method under consideration, e.g., SABR delta $\delta^{SABR}_{t_i,T_i,K_i}$ and Bartlett delta $\delta^{Bartlett}_{t_i,T_i,K_i}$. $\Delta V^{mkt}_{t_i,T_i,K_i}$ and ΔS_{t_i} are the changes in market option and underlying prices over a fixed time interval. Here we compare the **daily** local hedging risk with the daily changes of prices. The results are shown in Table 2.1: As we can see in Table 2.1, $\delta^{Bartlett}_{t,T,K}$ performs much better than $\delta^{SABR}_{t,T,K}$. What is more,

Method	SABR $\delta^{SABR}_{t,T,K}$	Bartlett $\delta^{Bartlett}_{t,T,K}$
Gain (%)	-4.2	27.1

Table 2.1: Daily local hedging risk comparison between original delta $\delta^{SABR}_{t,T,K}$ from SABR model and $\delta^{Bartlett}_{t,T,K}$ from Bartlett correction. The performance is evaluated in terms of improvement over Black-Scholes delta on local hedging risk.

$\delta^{SABR}_{t,T,K}$ actually performs worse than Black–Scholes delta $\delta^{BS}_{t,T,K}$ with implied volatility. The similar phenomenon has been observed in [123] where Heston and Heston-Nandi model perform worse than Black–Scholes delta $\delta^{BS}_{t,T,K}$ with implied volatility. **The difference of the hedging performance between $\delta^{Bartlett}_{t,T,K}$ and $\delta^{SABR}_{t,T,K}$ is an example to illustrate how the unaccounted pricing model parameter dependence on underlying asset affects the hedging performance.** Unfortunately, the pricing model parameter dependence often is not that easy to be accounted for as in SABR model with Bartlett correction.

Lastly, we introduce the corrective formula for minimum variance hedging based on SABR model implemented by Hull and White [112] . We use δ^{SV} to denote the minimum variance hedging based on SABR model from [112]. Let ΔF be a small changes in forward F, the δ^{SV} is given by:

$$\delta^{SV}_{t,T,K} = \frac{V_B(F_t + \Delta F, t, T, K, r; \sigma_B(F_t + \Delta F, t, T, K; \alpha, \beta, \nu, \rho)) - V_B(F_t, t, T, K, r; \sigma_B(F_t, t, T, K; \alpha, \beta, \nu, \rho))}{\Delta F} \tag{2.3.20}$$

with V_B be the Black pricing formula and σ_B be the SABR implied volatility formula discussed in section 2.2.3.

2.4 Motivation For Direct Nonparametric Data-Driven Hedging Models

In the previous sections, we discuss the issue of parameter dependence and present several correction methods under Black–Scholes framework. Although the corrective formula (2.3.5), (2.3.14), (2.3.17), and (2.3.18) adopt similar forms, the data-driven MV delta (2.3.5) is significantly different from other corrective formula for the fact that it is based on historical data while (2.3.14) ,(2.3.17), and (2.3.18) are based on pricing model calibrated to the spot options and underlying prices. Hull and White [112] indicate that MV delta $\delta^{MV}_{t,T,K}$ estimated based on historical data performs better than $\delta^{LVF}_{t,T,K}$ and $\delta^{SV}_{t,T,K}$ estimated based on spot data.

Our exploration on the data-driven models for hedging start roughly the same years as the work of [112] and we adopt the similar methodology as the MV delta where we learn a hedging position function from historical data. However, our explorations are motivated differently:

- The hedging position $\delta^{MV}_{t,T,K}$, $\delta^{LVF}_{t,T,K}$ and $\delta^{SV}_{t,T,K}$ proposed in [112] is to correct for the BS implied volatility dependence on underlying asset prices. Specifically, Hull and White [112] demonstrate how one can estimate the $\frac{\partial \sigma^{imp}}{\partial S}$ empirically.
- Noticing the pricing parameters dependence, we try to learn a hedging position directly from the market data. We are not specifically trying to correct parameters dependence for certain pricing models. We are trying to avoid it by obtaining hedging position without a pricing model at all.

In addition, the data-driven models proposed in this thesis and MV delta are also different in the following ways:

- Hull and White [112] assume a quadratic form to account for the implied volatility dependence on the underlying asset within Black–Scholes framework. The hedging position from our proposed data-driven models is purely determined by market data with machine learning algorithms with no specific parametric form.
- The MV hedging model (2.3.5) focuses on instantaneous hedging analysis (2.1.4). For discrete hedging, particularly when re-balancing is done infrequently, e.g., weekly or monthly, (2.3.5) may no longer be suitable. Our proposed models, which are not based on computing sensitivity of option pricing functions with regards to the underlying asset prices, can potentially improve the hedging results when the hedging is done less frequently.
- Our proposed data-driven models GRU_{δ} and $\text{GRU}_{\text{TOTAL}}$ can naturally include feature selection and feature extraction which can potentially improve the hedging performance.
- Our proposed data-driven model $\text{GRU}_{\text{TOTAL}}$ is enhanced to deal with total risk hedging scenarios instead of focusing on reducing the instantaneous local hedging risk as the MV hedging model (2.3.5).

In the following chapters, we will start to formally describe the proposed hedging models and present numerical hedging performance comparisons on both synthetic data and real market data.

Chapter 3

Data-Driven Kernel Learning Framework for Local Hedging Risk

In this chapter, we start the discussion on our proposed data-driven local hedging model from kernel learning framework. The data-driven kernel hedging model can be summarized as the following: Assume we have M data instances. Each observation of a market European option price $V^{mkt}_{t_i,K_i,T_i}$ is uniquely associated with a triplet $\{t_i, T_i, K_i\}$, where t_i is the trading time of the option price, K_i is the strike, and expiry T_i, $i = 1,\ldots,M$. The hedging position function is determined by the quadratic data-driven local risk minimization problem:

$$\min \frac{1}{2M}\sum_{i=1}^{M}\left(\Delta S_{t_i}\delta(\mathbf{x}_{t_i}^{T_i,K_i}) - \Delta V^{mkt}_{t_i,K_i,T_i}\right)^2$$

With ΔS_{t_i} and $\Delta V^{mkt}_{t_i,K_i,T_i}$ given in (2.1.1) for a fixed time interval Δt. The hedging position $\delta(\mathbf{x}_{t_i}^{T_i,K_i})$ is given by a data-driven hedging function $\mathrm{DKL}_{\mathrm{SPL}}(\mathbf{x}_{\mathbf{t}}^{\mathbf{T},\mathbf{K}};\widehat{\boldsymbol{\alpha}}^*)$:

$$\delta(\mathbf{x}_{t_i}^{T_i,K_i}) = \mathrm{DKL}_{\mathrm{SPL}}(\mathbf{x}_{\mathbf{t}}^{\mathbf{T},\mathbf{K}};\widehat{\boldsymbol{\alpha}}^*)$$

with $\widehat{\boldsymbol{\alpha}}^*$ being the parameters for the hedging functions and $\mathbf{x}_t^{T,K}$ being the input features for the hedging function. Please note that $\widehat{\boldsymbol{\alpha}}^*$ in this chapter is the parameter for the regularized kernel network to be learnt from data. The α in section 2.2.3 is the volatility for SABR. They are two unrelated notations.

From chapter 2, we have seen various challenges for hedging when the position is computed from the calibrated option value function. Since nonparametric option function estimation does not make specific assumptions and can potentially match option prices more accurately, it is not unreasonable to expect that this hedging challenge can potentially be addressed by reducing mis-specification using a data-driven approach to learn an option value function. Here we in this chapter we also discuss this approach and analyze its challenging for option hedging. Similar to discrete hedging under the parametric model, the hedging position can be computed by

learning a nonparametric option value model and then determining hedging position from the partial derivatives. Indeed this is the approach adopted in [114]. In this chapters, we indicate that the issue of pricing model parameters dependence still exists even if one estimate the pricing model using a machine learning model with no assumption on the dynamic of the underlying asset movement.

The organization of this chapter is as the following: We discuss how one can compute a hedging positions from partial derivatives of the optimal regularized kernel functions in section 3.1.1. The proposed data-driven kernel hedging learning method is presented in section 3.2. In section A, we discuss how cross-validation can be done efficiently for both kernel pricing and kernel hedging model. In section 3.3, we present experiments using synthetic data to illustrate the drawbacks of determining hedging position from partial derivative of pricing function estimated using machine learning algorithms and the effectiveness of data-driven direct kernel hedging functions.

3.1 Regularized Kernel Pricing Model

Recognizing various challenges in the parametric financial modelling approach, nonparametric option pricing has also been studied. The nonparametric option value modeling approach has the distinctive advantage of not relying on specific assumptions about the underlying asset price dynamics. Hutchinson et al. [114] first propose a nonparametric data-driven approach to price and hedge European options using neural networks, radial basis functions, and projection pursuit regressions. Many other neural network methods for European option pricing have also been proposed, see, e.g., [184, 17, 93, 82, 131].

Although there are quite a few studies on nonparametric option pricing models, to our knowledge, there has been little research specifically focusing on discrete hedging using a nonparametric method. Even when the hedging problem is considered, e.g., [114], it is treated as a byproduct of obtaining a nonparametric pricing function: The hedging position is obtained from the partial derivative of the option value function. Hutchinson et al. [114] show that, based on hedging errors on some simulated paths, this indirect data-driven hedging approach can potentially be an effective alternative to the traditional parametric delta hedging methods.

In this section, we follow the methodology of [114] and learn a data-driven option pricing model with the regularized kernel network. The hedging position is then given by the partial derivative of the option pricing model with regards to the underlying asset. Our goal is to learn a nonlinear option pricing function $V(\mathbf{x};\widehat{\boldsymbol{\alpha}}^*)$ using a regularized kernel method [73].

Assume that we are given a positive definite kernel similarity

$$\mathscr{K}(\mathbf{x},\mathbf{x}') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R},$$

which captures similarity between $\mathbf{x}$ and $\mathbf{x}'$ implicitly in a high dimensional feature space. Assume that $\mathscr{H}_K$ is the *Reproducing Kernel Hilbert Space* (RKHS) induced by the symmetric pos-

itive definite kernel function $\mathscr{K}(\mathbf{x},\mathbf{y})$ and $\|f\|_{\mathscr{K}}$ is the norm in RKHS. We have the following Representer Theorem [176]:

Theorem 3.1.1. *(The Representer Theorem) Let $\mathscr{X}$ be a nonempty set and k a positive definite real-valued kernel on $\mathscr{X}\times\mathscr{X}$ with the RKHS $\mathscr{H}_K$. Given the a training sample $\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_M\}$, a strictly monotonically increasing real valued function $g:[0,\infty)\to\mathbb{R}$, and an arbitrary empirical risk function $E:\mathscr{X}^M\to\mathbb{R}$ and E depends on f only through $\{f(\mathbf{x}_1),f(\mathbf{x}_2),\cdots,f(\mathbf{x}_M)\}$, then for any $f^*\in\mathscr{H}_K$ satisfying*

$$f^*=\underset{f\in\mathscr{H}_K}{\operatorname{argmin}}\ E\left(f(\mathbf{x}_1),f(\mathbf{x}_2),\cdots,f(\mathbf{x}_M)\right)+g(\|f\|_{\mathscr{K}})$$

f^* admits a representation of the form:

$$f^*(\cdot)=\sum_{i=1}^{M}\widehat{\alpha}_i^*\mathscr{K}(\cdot,\mathbf{x}_i)$$

with $\widehat{\alpha}_i^*\in\mathbb{R}$ for all $1\le i\le M$

A regularized kernel regression problem can be formulated as

$$\min_{f\in\mathscr{H}_K}\left(\sum_{i=1}^{M}L(f(\mathbf{x}_i))+\lambda_P\|f\|_{\mathscr{K}}^2\right)\tag{3.1.1}$$

where $L(\cdot)$ is a loss function. The regularization parameter $\lambda_P>0$ can be determined based on cross validation.

Following the Representer Theorem, e.g., [176], a solution of (3.1.1) has the form

$$f(\mathbf{x})=\sum_{i=1}^{M}\widehat{\alpha}_i^*\mathscr{K}(\mathbf{x},\mathbf{x}_i)\tag{3.1.2}$$

and the regularization term is given by

$$\|f\|_{\mathscr{K}}^2=\sum_{i=1}^{M}\sum_{j=1}^{M}\widehat{\alpha}_i^*\widehat{\alpha}_j^*\mathscr{K}(\mathbf{x}_i,\mathbf{x}_j).\tag{3.1.3}$$

Assume that a set of M training points $\{(\mathbf{x}_{t_1}^{T_1,K_1},V_{t_1,K_1,T_1}^{mkt}),\cdots,(\mathbf{x}_{t_M}^{T_M,K_M},V_{t_M,K_M,T_M}^{mkt})\}$ are given, where $\mathbf{x}_t^{T,K}\in\mathbb{R}^{d_l}$ is the input feature for pricing the option with strike K and expiry T at time t and $V_{t,K,T}^{mkt}$ is the market option price at time t with strike K and expiry T. Each data instance corresponds to a unique triplet $\{t,T,K\}$. We can estimate an option value function $V(\mathbf{x};\widehat{\boldsymbol{\alpha}}^*)$ based on the regularized kernel estimation (3.1.1) with $\widehat{\boldsymbol{\alpha}}^*=\{\widehat{\alpha}_1^*,\ldots,\widehat{\alpha}_M^*\}$.

Using (3.1.2) and (3.1.3), assuming quadratic loss, the option pricing function:

$$V(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}}^*) = \sum_{i=1}^{N} \widehat{\alpha}_i^* \mathscr{K}(\mathbf{x}_t^{T,K}, \mathbf{x}_{t_i}^{T_i,K_i}) \tag{3.1.4}$$

can be computed by solving

$$\widehat{\boldsymbol{\alpha}}^* = \underset{\alpha}{\operatorname{argmin}} \left(\sum_{i=1}^{M} \left(V_{t_i,T_i,K_i}^{mkt} - \sum_{j=1}^{M} \widehat{\alpha}_j \mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j}, \mathbf{x}_{t_i}^{T_i,K_i}) \right)^2 + \lambda_P \sum_{i=1}^{M} \sum_{j=1}^{M} \widehat{\alpha}_i \widehat{\alpha}_j \mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j}, \mathbf{x}_{t_i}^{T_i,K_i}) \right) \tag{3.1.5}$$

For standard options, the universal RBF kernel

$$\mathscr{K}(\mathbf{x}, \tilde{\mathbf{x}}) = e^{-\frac{\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2}{2\hat{\rho}^2}} \tag{3.1.6}$$

is a reasonable kernel choice, since the option value function is very smooth, and a suitable bandwidth $\hat{\rho}$ is typically problem dependent and can be determined using cross validation.

3.1.1 Indirect Hedging Positions From Kernel Pricing Functions

Assume that the one dimension of the attribute vector $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$ corresponds to the moneyness S_t/K at time t and other dimensions are not related to underlying prices. Then the delta hedging function option with strike K and expiry T at trading time t is typically determined as:

$$\delta_{t,T,K}^{IKL} = \frac{\partial V(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}}^*)}{\partial S} = \sum_{i=1}^{M} \widehat{\alpha}_i^* \frac{\partial \mathscr{K}(\mathbf{x}_t^{T,K}, \mathbf{x}_{t_i}^{T_i,K_i})}{\partial S} = \sum_{i=1}^{M} \frac{\widehat{\alpha}_i^*}{K} \frac{\partial \mathscr{K}(\mathbf{x}_t^{T,K}, \mathbf{x}_{t_i}^{T_i,K_i})}{\partial S/K} \tag{3.1.7}$$

Hutchinson et al. [114] demonstrate that this nonparametric hedging approach, using the partial derivative of a nonparametric pricing function learned from historical market data, can be a useful alternative for option hedging.

However, using (3.1.7) as the hedging position similarly does not minimize variance of hedge risk in general and the challenge in accounting for parameter dependence on the underlying remains. We can see that from the following arguments. We made an unrealistic assumption that the estimated kernel function $V(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}}^*)$ matches the target market option price exactly, i.e.,

$$V(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}}^*) = V_{t,T,K}^{mkt}$$

Then we have:

$$\sum_{i=1}^{M} \widehat{\alpha}_i^* \frac{\partial \mathscr{K}(\mathbf{x}_t^{T,K}, \mathbf{x}_{t_i}^{T_i,K_i})}{\partial S} + \sum_{i=1}^{M} \frac{\partial \widehat{\alpha}_i^*}{\partial S} \mathscr{K}(\mathbf{x}, \mathbf{x}_i) = \frac{\partial V^{mkt}}{\partial S}$$

Again, since model calibration only ensures matching in the option values, not matching the

change in the market option price, in general

$$\frac{\partial \widehat{\alpha}_i^*}{\partial S} \neq 0$$

unless

$$\frac{\partial V}{\partial S}(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*) = \frac{\partial V^{mkt}}{\partial S}. \tag{3.1.8}$$

However, there is no reason that a solution of the regression problem (3.1.5) should satisfy (3.1.8). Consequently it is similarly difficult to account for all dependence from $\widehat{\boldsymbol{\alpha}}^*$ on the underlying asset, even infinitesimally, in the estimated kernel model.

Furthermore, error magnification can happen by deriving the hedging position from a estimated kernel function. In general, the estimated kernel pricing function $V(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*)$ can not match the all the target market option prices exactly and the estimated kernel pricing function inevitably will have prediction error when used to predict the price of unobserved testing data instances. The calibration error and prediction can potentially increase the hedging error when using the $\frac{\partial V}{\partial S}(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*)$ as the hedging position.

Lastly, with above data-driven pricing approach, we predict the option value without considering the arbitrage constraints in the resulting price surfaces. As indicated by many recent studies [30, 192, 127], the option prices produced by data-driven machine learning model directly often contains arbitrage opportunities. Therefore, recent proposed data-driven pricing models [30, 192, 127] start to focus on predicting arbitrage-free implied volatility surface and the resulting volatility surfaces used together with the Black-Scholes model to produce the prices and hedging positions. As one can readily see, this data-driven pricing approach still omit the dependence of the implied volatility on the underlying asset prices since they focus on matching the observed historical implied volatility surfaces than matching the changes of implied volatility surfaces with regards to the underlying asset prices.

3.2 Regularized Kernel Hedging Model

Let us again assume that a set of M training points

$$\left\{ (\mathbf{x}_{t_1}^{T_1,K_1}, \Delta V_{t_1,K_1,T_1}^{mkt}, \Delta S_{t_1}), \cdots, (\mathbf{x}_{t_M}^{T_M,K_M}, \Delta V_{t_M,K_M,T_M}^{mkt}, \Delta S_{t_M}) \right\}$$

are given, where $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$ is the input feature for **hedging** the option with strike K and expiry T at time t. The $\Delta V_{t,K,T}^{mkt}$ is the change of market option price at time t with strike K and expiry T over a fixed time intervel Δt, e.g., daily, weekly, or monthly. The ΔS_t is the change of underling price at time t over over the same Δt. $\Delta V_{t,K,T}^{mkt}$ and ΔS_t are given by equation (2.1.1). Again, each data instance corresponds to a unique triplet $\{t, T, K\}$.

We can estimate an option hedging function $\delta(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*)$ based on the regularized kernel network with $\widehat{\boldsymbol{\alpha}}^* = \{\widehat{\alpha}_1^*, \ldots, \widehat{\alpha}_M^*\}$. The empirical loss function is chosen to correspond to the

square of discrete local hedging risk in section 2.1, i.e.,

$$L\left(\delta(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}})\right) = \left(\Delta V_{t,K,T}^{mkt} - \Delta S_t \delta(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}})\right)^2. \tag{3.2.1}$$

The hedging position function $\delta(\mathbf{x}_t^{T,K};\widehat{\boldsymbol{\alpha}}^*)$ can be estimated from the regularized optimization below:

$$\min_{\delta\in\mathscr{H}_K}\left\{\sum_{i=1}^{M} L\left(\delta(\mathbf{x}_{t_i}^{T_i,K_i};\widehat{\boldsymbol{\alpha}})\right)^2 + \lambda_P\|\delta\|_{\mathscr{K}}^2\right\} \tag{3.2.2}$$

Note that the Representer Theorem still holds for (3.2.2). Using the Representer Theorem (3.1.2) and (3.1.3), (3.2.3) can be computed by solving the following convex quadratic minimization,

$$\widehat{\boldsymbol{\alpha}}^* = \underset{\widehat{\boldsymbol{\alpha}}}{\operatorname{argmin}}\left(\sum_{i=1}^{M}\left(\Delta V_{t_i,T_i,K_i}^{mkt} - \Delta S_{t_i}\sum_{j=1}^{M}\widehat{\alpha}_j\mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j},\mathbf{x}_{t_i}^{T_i,K_i})\right)^2 + \lambda_P\sum_{i=1}^{M}\sum_{j=1}^{M}\widehat{\alpha}_i\widehat{\alpha}_j\mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j},\mathbf{x}_{t_i}^{T_i,K_i})\right) \tag{3.2.3}$$

The loss function for the hedging function in the proposed formulation (3.2.2) directly corresponds to the sum of squares of local hedging risk for a discrete Δt-time period. In addition the hedge function is the solution of the optimization problem and there is no potential error magnification through partial derivative computation. We avoid the pricing model parameters dependence on underlying asset by not computing a prcing model at all.

Since at the expiry the delta of the payoff function is a discontinuous step function, the delta hedging function of an option changes quickly as the underlying changes near the expiry. Consequently we choose to use a spline kernel function [174] for the hedging function estimation. The explicit expression for the spline kernel with order O_d for the one-dimensional case is :

$$\mathscr{K}(x,y) = \sum_{r=0}^{O_d}\frac{\binom{O_d}{r}}{2O_d - r + 1} min(x,y)^{2O_d-r+1}|x-y|^r + \sum_{r=0}^{O_d} x^r y^r$$

For multidimensional data, the spline kernel is the product of one-dimensional spline kernel functions with respect to each dimension. Interested readers can referred to [174] for more details. The main benefit of using the spline kernel over the Gaussian kernel is that spline kernel does not have a hyperparameter to be tuned while Gaussian kernel has a bandwidth parameter to be tuned by cross-validation which is often costly. In this thesis, we focus on spline kernel and, in section 3.3, we will show that spline kernel performs better or equally well when compared with Gaussian kernel. In the latter discussion, we denote $\delta_{t,T,K}^{DKL} = \delta(\mathbf{x}_t^{T,K}\widehat{\boldsymbol{\alpha}}^*)$ as the hedging position form the direct kernel hedging model. For our data-driven approach, we need to select an appropriate penalty λ_P to control the model complexity. Cross-validation (CV) is a commonly used method for the performance estimation and model selection for the learning algorithms. For example, the Leave-One-Out Cross-Validation (LOOCV) computes the output for each data instance using parameters trained on the remaining data instances. For the regularized kernel methods, we can compute the CV error efficiently without retraining the model in each CV

round [148]. More detailed discussion on the fast LOOCV of regularized kernel network [148] can be found in appendix A.

3.3 Comparison using synthetic data

Following the S&P 500 market option specifications in [113], we first synthetically generate option data assuming that the underlying price follows a Heston model [103]. We compare hedging effectiveness of direct hedging function learning (3.2.2) and indirect hedging function estimation (3.1.5). In addition, we compare their performance to that of using analytic delta under the Heston model, which can be regarded as a best case benchmark, at least for daily hedging. Note that there is no model mis-specification for the underlying price in this synthetic case.

The experiments in the this section is to demonstrate the following :

- Computing the hedging position as the partial derivative of an nonparametric option pricing model with regards to underlying asset still suffers from the issue of parameter dependence on underlying asset.
- The hedging position directly learnt from data as in (3.2.3) can be very close to the best case benchmark in daily hedging.
- The hedging position directly learnt from data using spline kernel performs better or equally well when compared with Gaussian kernel.

Since the loss function is quadratic, solutions to (3.2.2) and (3.1.5) can be easily computed from linear equation solvers. In addition we also compare hedging performance using a RBF kernel (3.1.6) versus a spline kernel. Specifically, using the generated synthetic data, we compare here hedging performance of the following hedging computation methods:

- $\delta^{BS}_{t,T,K}$: implied volatility BS delta
- $\delta^{Heston}_{t,T,K}$: analytical Heston delta
- DKL_{SPL} : directly learning a spline kernel hedging function based on (3.2.2)
- DKL_{RBF} : directly learning a RBF kernel hedging function directly based on (3.2.2)
- IKL_{SPL}: determining hedging position indirectly as the partial derivative (3.1.7) of the option value function estimated from (3.1.5) using a spline kernel
- IKL_{RBF}: determining hedging position indirectly as the partial derivative (3.1.7) of the option value function estimated from (3.1.5) using a RBF kernel

Training data consists of simulated daily underlying price S_t for two years[1], $t = 1, \cdots, 2 \times 252$, assuming a risk-neutral Heston model below:

$$\begin{aligned} dS &= (r-q)Sdt + \sqrt{\Upsilon} S d\hat{W} \\ d\Upsilon &= \kappa^*(\overline{\Upsilon}^* - \Upsilon)dt + \eta\sqrt{\Upsilon} d\hat{Z} \\ E[d\hat{Z}d\hat{W}] &= \rho dt \end{aligned}$$

In addition, a vector of simulated (traded) market call option prices are generated, on each day t, with different strikes and time to expiry, following the CBOE specifications of stock options described in [113]. The option prices V^{mkt} are computed using the analytical option formula (2.2.10) under the Heston model [103] in section 2.2.2 , using the the parameters from [13], which are given in Table 3.1. In other words, we assume in this section:

$$V^{mkt}_{t,T,K} = V_{Heston}(S_t, t, T, K, r, q; \Upsilon, \kappa^*, \overline{\Upsilon}^*, \eta, \rho)$$

Note that the Heston delta position under this synthetic scenarios does not have the parameter dependence issue, since for the synthetic case all heston parameter are fixed, and can be used as the benchmark.

$$\delta^{Heston}_{t,T,K} = \frac{\partial V_{Heston}(S_t, t, T, K, r, q; \Upsilon, \kappa^*, \overline{\Upsilon}^*, \eta, \rho)}{\partial S} = \frac{\partial V^{mkt}}{\partial S}$$

The Black-Scholes delta position from implied volatility still has the the issue of implied volatility depending on underlying asset:

$$\delta^{BS}_{t,T,K} = \frac{\partial V_{BS}(S_t, t, T, K, r, q; \sigma^{imp}_{t,T,K})}{\partial S} \neq \frac{\partial V^{mkt}}{\partial S}$$

where $\sigma^{imp}_{t,T,K}$ is the volatility that equals Black–Scholes price and Heston price

$$V_{BS}(S_t, t, T, K, r, q; \sigma^{imp}_{t,T,K}) = V_{Heston}(S_t, t, T, K, r, q; \Upsilon, \kappa^*, \overline{\Upsilon}^*, \eta, \rho)$$

r	q	$\overline{\Upsilon}^*$	κ^*	η	ρ	S_0	Υ_0
0.02	0.0	0.04	1.15	0.39	-0.64	100	0.04

Table 3.1: Parameters for the Heston model. This set of parameters is used to generate synthetic experiments for demonstrating the effectiveness of DKL_{SPL} on local risk hedging.

Testing data consists of 100 daily underlying price paths and corresponding option prices, spanning a six month period. The Heston parameter is specified as in Table 3.1. We report

[1] We assume that there are 252 trading days in a year.

average performance measures over 10 random training-test-data sets generated as described. [2]

We train a regularized option price kernel model for the indirect hedging IKL_{SPL} and IKL_{RBF}. For IKL_{SPL} and IKL_{RBF}, the hedging position is the partial derivative of the estimated pricing function (3.1.7), as in [114]. We note that the simulation hedging analysis in [114] considers only the Black–Scholes model and the results in [114] indicate that a lower hedging error can be achieved on a subset of simulated paths.

We use the standard deviation of the pairwise Euclidean distance of the training data as the bandwidth ρ for the RBF kernel. The regularization parameter λ_P is however selected using a 5-fold cross-validation. We also consider weekly hedging and monthly hedging, which correspond to hedging over a 5-business-days period and 20-business-days period respectively.

To investigate impact of the feature choice, we evaluate hedging performance using

- Feature Set #1 $= \{\text{MONEYNESS } (S_t/K),\ \text{TIME-TO-EXPIRY } (T-t)\}$.
- Feature Set #2 $= \{\text{MONEYNESS } (S_t/K),\ \text{TIME-TO-EXPIRY } (T-t),\ \ \delta^{BS}_{t,T,K}\}$.

In the second feature set, the Black–Scholes delta $\delta^{BS}_{t,T,K}$ using the implied volatility is used as an additional feature in determining the hedging position.

Let the number of data instances to be evaluated be m. As in section 2.1.1, the local hedging risk over the fixed time interval Δt for each data instance, which corresponds to a unique triplet $\{t,T,K\}$, is given as below:

$$\text{Risk}^{local}_{t,T,K} = \Delta S_t \delta_{t,T,K} - \Delta V^{mkt}_{t,K,T}$$

where $\delta_{t,T,K}$ is the hedging position from different approaches, e.g., Black–Scholes delta $\delta^{BS}_{t,T,K}$, Heston delta $\delta^{Heston}_{t,T,K}$, direct data-driven hedging position $\delta^{DKL}_{t,T,K}$, and indirect data-driven hedging position$\delta^{IKL}_{t,T,K}$. We evaluate the hedging performance in four different ways:

1. Gain (2.3.19) over Black-Scholes $\delta^{BS}_{t,T,K}$ as in [112].
2. The mean absolute value of $\text{Risk}^{local}_{t,T,K}$

$$\mathbf{E}(|\Delta V^{mkt} - \Delta S\delta|) = \frac{1}{m}\sum_{i=1}^{m} |\text{Risk}^{local}_{t_i,T_i,K_i}|$$

3. The 95% Value-at-Risk (VaR) of $\{\text{Risk}^{local}_{t_i,T_i,K_i} | i = 1,\ldots,m\}$
4. The 95% Conditional-Value-at-Risk (CVaR) of $\{\text{Risk}^{local}_{t_i,T_i,K_i} | i = 1,\ldots,m\}$

[2] Following CBOE option specification rules, the size of a training or testing data set can vary slightly for each simulation run.

3.3.0.1 Feature Set #1:{MONEYNESS (S_t/K), TIME-TO-EXPIRY $(T-t)$}

For the synthetic data, it is known that the option price is a function of the moneyness $\frac{S}{K}$ and time to expiry $T-t$, which are the attributes $\mathbf{x}_t^{T,K}$ in Feature Set #1. Table 3.2, 3.3 and 3.4 report results for daily, weekly, and monthly hedging respectively.

Method	Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
δ_{BS}	0.0	0.185	0.286	0.380	0.574
IKL_{RBF}	-3.3	0.171	0.291	0.356	0.566
IKL_{SPL}	-183.3	0.291	0.482	0.669	1.105
DKL_{RBF}	63.1	0.120	0.174	0.251	0.352
DKL_{SPL}	**64.9**	**0.121**	**0.170**	**0.255**	**0.345**
HESTON	63.6	0.121	0.173	0.266	0.360

Table 3.2: Daily hedging comparison on synthetic experiments between various hedging strategies. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #1: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY}\}$
[2] Bold entry indicating best Gain

Method	Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
δ_{BS}	0.0	0.414	0.620	0.776	1.009
IKL_{RBF}	-197.6	0.406	1.070	0.741	1.254
IKL_{SPL}	-94.7	0.548	0.866	1.114	1.738
DKL_{RBF}	47.0	0.312	0.451	0.620	0.825
DKL_{SPL}	**50.8**	**0.312**	**0.435**	**0.622**	**0.797**
HESTON	45.7	0.319	0.456	0.651	0.840

Table 3.3: Weekly hedging comparison on synthetic experiments between various hedging strategies. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #1: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY}\}$
[2] Bold entry indicating best Gain

Table 3.2 and 3.3 demonstrate that the direct hedging function learning DKL_{SPL} & DKL_{RBF} significantly outperform the indirect hedging learning IKL_{SPL} and IKL_{RBF} in Gain and different risk measures considered. Indeed, DKL_{SPL} and DKL_{RBF} slightly outperform the benchmark of using the analytic Heston delta. The indirect hedging function learning performs more poorly than the implied BS delta hedging. In addition, the spline kernel performs better than the RBF kernel (with the standard deviation as the bandwidth parameter). The RBF kernel yields larger risk measures and smaller Gain for both the direct and indirect hedging learning methods.

Method	Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
δ_{BS}	0.0	0.941	1.484	1.516	1.808
IKL_{RBF}	1.0	0.888	1.470	1.516	1.829
IKL_{SPL}	-36.9	1.135	1.729	2.033	2.894
DKL_{RBF}	33.6	0.860	1.181	1.612	1.949
DKL_{SPL}	35.4	0.858	1.165	1.610	1.922
HESTON	**38.7**	**0.814**	**1.136**	**1.544**	**1.829**

Table 3.4: Monthly hedging comparison on synthetic experiments between various hedging strategies. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #1: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY}\}$
[2] Bold entry indicating best Gain

Table 3.4 reports hedging comparison for monthly hedging. We observe that DKL_{SPL} & DKL_{RBF} significantly outperform the indirect hedging learning IKL_{SPL} & IKL_{RBF} in Gain and various risk measures. In addition, DKL_{SPL} & DKL_{RBF} continue to achieve enhanced performance over δ_{BS}, with the spline kernel DKL_{SPL} yielding better results than DKL_{RBF}. Not surprisingly, hedging performance of each method also deteriorates as the length of the hedging period increases, with larger mean absolute hedging error and larger standard deviation for monthly hedging than for daily and weekly hedging.

Table 3.4 also illustrates that, unlike daily and weekly hedging, DKL_{SPL} & DKL_{RBF} slightly underperform the analytic Heston delta benchmark for monthly hedging. Given that the analytic delta is for instantaneous hedging while the direct hedging learning DKL_{SPL} minimizes quadratic hedging error, one would expect better performance from the direct hedging learning DKL_{SPL}. We suspect that this is due to the effect of the specific combination of choices of features and kernel. Next we show that, with a different feature set, performance of direct hedging is improved, which suggests the possibility of surpassing analytic Heston delta benchmark, using a more suitable feature set, for a longer period hedging.

3.3.0.2 Feature Set #2: $\{\textbf{MONEYNESS}\ (S_t/K),\ \textbf{TIME-TO-EXPIRY}\ (T-t),\ \ \delta^{BS}_{t,T,K}\}$

We add the Black–Scholes delta $\delta^{BS}_{t,T,K}$ using the implied volatility as an additional feature in the direct hedging learning, since Hull and White [112] indicate that a better minimum variance hedge can be calculated based on the implied volatility delta. Table 3.5, 3.6, and 3.7 present hedging results for DKL_{SPL} & DKL_{RBF} for $\{\text{MONEYNESS}\ (\frac{S_t}{K}),\ \text{TIME-TO-EXPIRY}\ (T-t),\ \ \delta^{BS}_{t,T,K}\}$. For clarity, we also include the results for FS #1 $\{\text{MONEYNESS}\ (\frac{S_t}{K}),\ \text{TIME-TO-EXPIRY}\ (T-t)\}$ and Heston delta for ease of comparison.

From Table 3.5, 3.6 and 3.7, we observe that, for daily and weekly hedging, including the BS delta further improves the performance of the direct hedging learning methods, which out-

Method		Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
DKL_{RBF}	FS #1	63.1	0.120	0.174	0.251	0.352
	FS #2	62.3	0.114	0.176	0.238	0.349
DKL_{SPL}	FS #1	64.9	0.121	0.170	0.255	0.345
	FS #2	**70.9**	**0.110**	**0.154**	**0.234**	**0.322**
HESTON		63.6	0.121	0.173	0.266	0.360

Table 3.5: Daily hedging comparison on synthetic experiments between various hedging strategies. One additional feature δ_{BS} is added. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #2: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY, } \delta_{\text{BS}}\}$

[2] Bold entry indicating best Gain

Method		Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
DKL_{RBF}	FS #1	47.0	0.312	0.451	0.620	0.825
	FS #2	51.4	0.301	0.432	0.611	0.816
DKL_{SPL}	FS #1	50.8	0.312	0.435	0.622	0.797
	FS #2	**53.5**	**0.299**	**0.422**	**0.606**	**0.794**
HESTON		45.7	0.319	0.456	0.651	0.840

Table 3.6: Weekly hedging comparison on synthetic experiments between various hedging strategies. One additional feature δ_{BS} is added. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #2: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY, } \delta_{\text{BS}}\}$

[2] Bold entry indicating best Gain

Method		Gain (%)	$\mathbf{E}(\lvert\Delta V^{mkt} - \Delta S\delta\rvert)$	Std	VaR	CVaR
DKL_{RBF}	FS #1	33.6	0.860	1.181	1.612	1.949
	FS #2	30.1	0.863	1.217	1.652	2.104
DKL_{SPL}	FS #1	35.4	0.858	1.165	1.610	1.922
	FS #2	36.6	0.836	1.156	1.609	1.953
HESTON		**38.7**	**0.814**	**1.136**	**1.544**	**1.829**

Table 3.7: Monthly hedging comparison on synthetic experiments between various hedging strategies. One additional feature δ_{BS} is added. The hedging performance is evaluated in terms of local hedging risk.

[1] FS #2: $\mathbf{x} = \{\text{MONEYNESS, TIME-TO-EXPIRY, } \delta_{\text{BS}}\}$

[2] Bold entry indicating best Gain

performs analytical delta hedging. For monthly hedging, however, the performance is similar to what we obtain with that of the feature set #1. Overall, including the BS delta as an attribute is beneficial for the direct hedging function learning.

3.4 Enhancement Over the Kernel Local Hedging Model

In this chapter, we have illustrated that, even in a nonparametric kernel approach to model the option value function, dependence on the underlying can exist for the estimated kernel parameters. Consequently, using the partial derivatives of the model option pricing function, parametrically or nonparametrically estimated, will fail to minimize hedging error, even instantaneously.

Thus, we propose to directly learn nonparametric kernel hedging functions by minimizing the sum of square of the discrete local hedging risk, bypassing the intermediate step of the option value function estimation. Using synthetic data, we first demonstrate that the proposed direct hedging function learning significantly outperforms hedging based on the sensitivity of the model option function learned nonparametrically. In addition, we demonstrate that spline kernel yields better hedging performance in comparison to that of the RBF kernel. The exploratory research in this chapter clearly demonstrates the potential role of a market data-driven approach for financial derivative modelling and risk management.

However, we also notice the positive gain ratios (2.3.19) are smaller for monthly hedging in comparison to daily and weekly hedging for the synthetic data. In addition, when testing on real market S&P 500 index option with the DKL_{SPL} model and the feature set #2, we observed the similar phenomenon as it is for the synthetic scenario: the gain ratio decreases when we move from daily hedging to weekly and monthly hedging. The details of the real data comparison can be found in [143] and will be discussed in details in chapter 5.

We suspect that neither the feature set #1 nor the feature set #2 is enough to learn a sufficient data-driven local hedging model for longer period such as weekly and monthly. We are thus motivated to enhance the data-driven local hedging model using the RNN framework by adding the following components to include more features in computing the hedging position:

- Feature selection process.
- Sequential feature extraction process.

We demonstrate using real S&P 500 index option data that such enhancement greatly improve the local hedging performance. Besides, the GRU_δ proposed in Chapter 4 is more computational efficient enabling us to update the model more frequently to incorporate market changes. Details of the enhanced data-driven local hedging model GRU_δ will be discussed in chapter 4. The experimental results of DKL_{SPL} and GRU_δ on real $S\&P500$ index option data will be discussed together in chapter 5.

Chapter 4

Data-Driven Sequential Learning Framework for Local Hedging Risk

In chapter 3, we have explored the data-driven kernel hedging model DKL_{SPL} [143]. However, we believe that the data-driven hedging learning approach in [143] can potentially benefit from further improvements in a few directions. Firstly, Nian et al. [143] only use moneyness, time-to-expiry, and Black–Scholes delta as features to predict the hedging position. In practice, since both the underlying and options markets have complex price dynamics, feature extraction and feature selection can potentially further enhance hedging performance. Secondly, when predicting the hedging position at the rebalancing time t, only attributes observed at t are used as features in [143]. However, a financial market exhibits volatility clustering, describing a positive, significant, and slowly decaying volatility autocorrelation [132]. For example, a generalized autoregressive conditionally heteroskedastic (GARCH) model has been proposed for option pricing because of its capability in better characterizing asset returns [104] . It has been shown that the option pricing function under a GARCH model depends not only on the current underlying price but also on the observed underlying price history [62, 104]. This suggests that the information immediately prior to the rebalancing time should be relevant in determining the hedging position at the rebalancing time. Thirdly, the mean squared error is used as the loss function in [143] but a more appropriate robust objective function and framework may lead to a more stable optimal learning as the market shifts between crisis and normal regimes. Lastly, when a kernel data-driven model needs to be updated frequently, e.g., daily, it is computationally prohibitive to conduct backtesting over a long time period (e.g., a decade). Consequently the regularized spline kernel network is only updated monthly in [143]. To accurately assess the potential of a data-driven hedging approach, a model which is sufficiently computationally efficient needs to be considered to allow more frequent updating in a back-testing study of a long time horizon.

To incorporate sequential information in the hedging model, we use the Recurrent Neural Network (RNN), a feed-forward neural networks augmented with edges that connect adjacent steps. In the early 1980s, Hopfield [108] introduced RNN for sequential pattern recognitions. Jordan [119] and Elman [70] developed basic architectures for RNN with a group of neural

networks that have a "memory" to capture past information. This can be beneficial in time series applications.

Training RNN is similar to training the traditional neural network, with the Stochastic Gradient Descent (SGD) as the primary optimization tool. The Back Propagation Through Time (BPTT) algorithm is used to calculate gradients. However, vanishing and exploding gradients [107] can occur when back-propagating errors across many steps, which pose challenges for standard RNN architectures to learn long-term dependence between steps. The Long Short-Term Memory (LSTM) [106] model and the Gated Recurrent Unit (GRU) model [40] are subsequently proposed to address issues of vanishing and exploding gradients. LSTM and GRU utilize a gate structure with element-wise operations, which can retain information in memory for a longer period. This alleviates the problem of vanishing and exploding gradients [106]. GRU and LSTM models are shown to perform better than the standard RNN model [40], although performances of GRU and LSTM are comparable. Since GRU has fewer parameters than LSTM and usually require less training data [187], we use the GRU in the proposed encode-decoder hedging model GRU_δ.

4.1 The Proposed GRU_δ for market data-driven hedging

4.1.1 Local Discrete Hedging GRU Model

Figure 4.1 depicts the proposed local discrete GRU hedging model GRU_δ, which uses an encoder-decoder structure to combine the local and sequential features, similar to the sequence to sequence (seq2seq) models [40]. This model uses both the local features at the hedging time and the sequential features, which encode information immediately before the hedging time.

Consider a data instance at the hedging time t, strike K, and expiry T. Let the associated change of the market option price be $\Delta V^{mkt}_{t,T,K}$ and underlying price change be ΔS_t, as in (2.1.1). Assume that at the hedging time t, we have local features vector $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$ which records local information at the hedging time t for hedging the option with expiry T and strike K.

Let Δt_d denote the time interval for sequential information recording. In the subsequent empirical study, the interval Δt_d equals one-day. We denote the sequential features recording the daily history for hedging the option with expiry T and strike K as

$$\mathbf{Y}_t^{T,K} = \left[\mathbf{y}_{t-N\Delta t_d}^{T,K}, \ldots, \mathbf{y}_t^{T,K}\right]$$

For notational simplicity, we denote $\breve{\mathrm{t}}_i = t - (N+1-i)\Delta t_d$ with $i = 1, \ldots, N+1$, we thus have:

$$\mathbf{Y}_t^{T,K} = \left[\mathbf{y}_{\breve{\mathrm{t}}_1}^{T,K}, \ldots, \mathbf{y}_{\breve{\mathrm{t}}_{N+1}}^{T,K}\right]$$

The vector $\mathbf{y}_{\breve{\mathrm{t}}_i}^{T,K} \in \mathbb{R}^{d_s}$ has d_s features at time $\breve{\mathrm{t}}_i$ in the input sequential feature. Thus d_l is the

dimension for the local feature $\mathbf{x}_t^{T,K}$, d_s is the dimension for the sequential feature $\mathbf{Y}_t^{T,K}$, and $N+1$ is the length of the sequential feature sequence.

The encoder transforms information from the sequential feature $\mathbf{Y}_t^{T,K}$ to a fixed-sized vector $\widehat{\mathbf{h}}_E$ and the decoder makes the final prediction based on both $\widehat{\mathbf{h}}_E$ and the local feature $\mathbf{x}_t^{T,K}$. The overall structure of the proposed model is illustrated in Figure 4.1. Next we discuss each component of the proposed encoder-decoder model in details.

4.1.2 Feature Selection via Embedded Feature Weighting

Feature selection improves machine learning performance by eliminating noise and providing better interpretability. While various feature selection frameworks have been proposed, we consider the feature weighting method [172], which embeds feature selection in the SVM training. This embedded feature weighting method is shown to outperform other state-of-the-art embedded feature selection methods [172]. We adopt a similar feature weighting embedding technique in the discrete GRU model to conduct feature selections on both the local feature $\mathbf{x}_t^{T,K}$ and the sequential feature $\mathbf{Y}_t^{T,K}$. Features are first weighted and then fed into the encoder and decoder as inputs. Feature weights are optimized during model training.

We use a softmax function to generate a normalized feature weighting vector. For the local feature $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$, the j^{th} component of the normalized weight vector is given by

$$\frac{exp(\omega_j^L)}{\sum_{i=1}^{d_l} exp(\omega_i^L)}$$

The weighted local feature vector is defined as

$$\widehat{\mathbf{x}}_t^{T,K} = \frac{exp(\omega^L)}{\sum_{i=1}^{d_l} exp(\omega_i^L)} \odot \mathbf{x}_t^{T,K}$$

where $\odot$ denotes the element-wise multiplication.

Similarly, for the sequential feature $\mathbf{y}_{\breve{\mathrm{t}}_i}^{T,K}$, the j_{th} component of the normalized weight vector is given by

$$\frac{exp(\omega_j^S)}{\sum_{i=1}^{d_s} exp(\omega_i^S)}$$

The weighted feature vector at time $\breve{\mathrm{t}}_i$ is defined as

$$\widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K} = \frac{exp(\omega^S)}{\sum_{j=1}^{d_s} exp(\omega_j^S)} \odot \mathbf{y}_{\breve{\mathrm{t}}_i}^{T,K}$$

The weighting procedure acts as a feature selection. If a particular feature is not relevant in predicting the hedging position, the associated weight after learning is expected to be negligible.

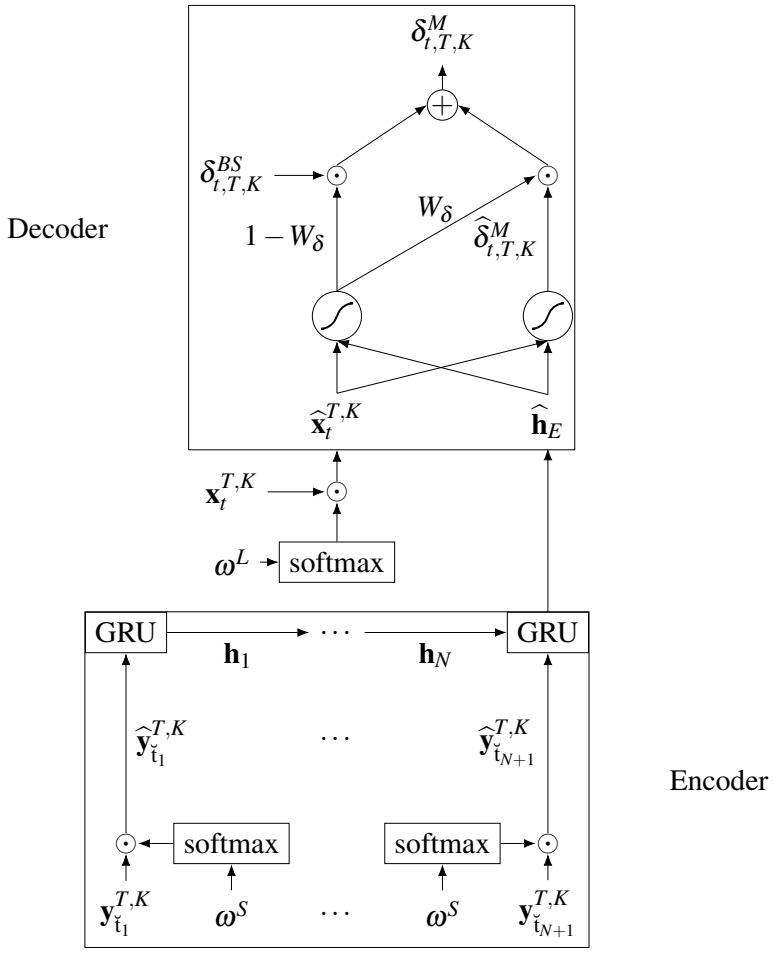


Figure 4.1: GRU_δ: GRU encode-decoder hedging model. The encoder summarizes the time series $\mathbf{Y}^{T,K}_t = \left[\mathbf{y}^{T,K}_{\check{t}_1}, \ldots, \mathbf{y}^{T,K}_{\check{t}_{N+1}}\right]$ as a succinct vector $\widehat{\mathbf{h}}_E$. The decoder outputs the hedging position based on the vector $\widehat{\mathbf{h}}_E$ and the local feature vector $\mathbf{x}^{T,K}_t$ observed at the hedging time t. More specifically, in the decoder, a candidate output $\widehat{\delta}^M_{t,T,K}$ is firstly produced. The final output $\delta^M_{t,T,K}$ is computed based on the linear combination of BS delta $\delta^{BS}_{t,T,K}$ and the candidate output $\widehat{\delta}^M_{t,T,K}$. The combination weight is determined by W_δ. The feature weight ω^L and ω^S are used to produce the weighted local feature and $\widehat{\mathbf{x}}^{T,K}_t$ and weighted sequential feature $\widehat{\mathbf{y}}^{T,K}_{\check{t}}$ respectively. The weighting acts as a feature selection process. Each edge in the graph has an arrow on it, pointing from a node whose output is used by the node pointed by the arrow as an input.

4.1.3 GRU Encoder

The proposed GRU_δ in Figure 4.1 has an one-layer GRU, which encodes the sequential feature $\mathbf{Y}_t^{T,K}$ to a fixed-sized vector $\widehat{\mathbf{h}}_E$. At the step i, the encoder computes the value of the hidden state $\mathbf{h}_i$ using a GRU cell. The input at the step i of the encoder is $\widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K}$, $i = 1,\ldots,N+1$. The internal structure of the GRU cell is shown in Figure 4.2.

Let $\boldsymbol{W}_z, \boldsymbol{U}_z, \mathbf{b}_z, \boldsymbol{W}_r, \boldsymbol{U}_r, \mathbf{b}_r, \boldsymbol{W}_h, \boldsymbol{U}_h, \mathbf{b}_h$ denote parameters shared by all GRU cells.

1. The **update gate** decides how much the cell updates its activation:

$$\mathbf{z}_i = sigmoid(\boldsymbol{W}_z \widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K} + \boldsymbol{U}_z \mathbf{h}_{i-1} + \mathbf{b}_z)$$

2. The **reset gate** decides how much information to retain from the previous hidden state $\mathbf{h}_{i-1}$:

$$\mathbf{r}_i = sigmoid(\boldsymbol{W}_r \widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K} + \boldsymbol{U}_r \mathbf{h}_{i-1} + \mathbf{b}_r)$$

3. The **candidate** hidden state value $\widehat{\mathbf{h}}_i$ is computed from the current input $\widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K}$, previous hidden state $\mathbf{h}_{i-1}$, and the reset value $\mathbf{r_i}$:

$$\widehat{\mathbf{h}}_i = tanh(\boldsymbol{W}_h \widehat{\mathbf{y}}_{\breve{\mathrm{t}}_i}^{T,K} + \boldsymbol{U}_h(\mathbf{r}_i \odot \mathbf{h}_{i-1}) + \mathbf{b}_h)$$

4. The **output** hidden state value $\mathbf{h}_i$ is computed based on a weighted combination of the previous activation $\mathbf{h}_{i-1}$ and the candidate activation $\widehat{\mathbf{h}}_i$:

$$\mathbf{h}_i = (1 - \mathbf{z}_i) \odot \mathbf{h}_{i-1} + \mathbf{z}_i \odot \widehat{\mathbf{h}}_i$$

The hidden state at the last step $\mathbf{h}_{N+1}$, corresponding to time $\breve{\mathrm{t}}_{N+1} = t$, is supplied to the decoder as the fixed size vector $\widehat{\mathbf{h}}_E$, which extracts relevant information in $\mathbf{Y}_t^{T,K}$.

4.1.4 Decoder

The decoder combines the output of the encoder from the sequential feature $\mathbf{Y}_t^{T,K}$ at the hedging time t with the current local feature $\mathbf{x}_t^{T,K}$. It uses one neural network to firstly compute a candidate output $\widehat{\delta}_{t,T,K}^M$, based on both the weighted local input $\widehat{\mathbf{x}}_t^{T,K}$ and the fixed size vector $\widehat{\mathbf{h}}_E$:

$$\widehat{\delta}_{t,T,K}^M = sigmoid(\mathbf{v}_{out}^T \, tanh(\boldsymbol{U}_{out} \widehat{\mathbf{h}}_E + \boldsymbol{W}_{out} \widehat{\mathbf{x}}_t^{T,K} + \mathbf{b}_{out})).$$

The the output gate value W_δ is given by another neural network, based on both the weighted local input $\widehat{\mathbf{x}}_t^{T,K}$ and the fixed size vector $\widehat{\mathbf{h}}_E$:

$$W_\delta = sigmoid(\mathbf{v}_{Gate}^T \, tanh(\boldsymbol{U}_{Gate} \widehat{\mathbf{h}}_E + \boldsymbol{W}_{Gate} \widehat{\mathbf{x}}_t^{T,K} + \mathbf{b}_{Gate})). \tag{4.1.1}$$

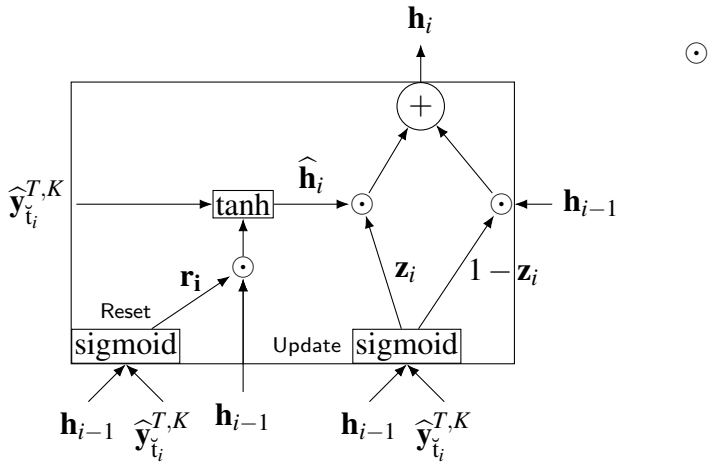


Figure 4.2: Illustration of the GRU cell. At step i, GRU cell produces a vector $\mathbf{h}_i$ as the current hidden states based on the hidden states produced from the previous steps $\mathbf{h}_{i-1}$ and the input $\widehat{\mathbf{y}}^{T,K}_{t_i}$. The reset unit produces the weight $\mathbf{r}_i$ which determines how much information is retained from $\mathbf{h}_{i-1}$. The final output from the GRU cell $\mathbf{h}_i$ is computed based on a weighted combination of the previous activation $\mathbf{h}_{i-1}$ and the candidate activation $\widehat{\mathbf{h}}_i$, where the combination weight $\mathbf{z}_i$ is produced by the update unit. Each edge in the graph has an arrow on it, pointing from a node whose output is used by the node pointed by the arrow as an input.

The practitioner's BS delta encapsulates a significant portion of the sensitivity of the option price to the underlying and has been adopted in option hedging practice. Therefore we use BS implied volatility delta as a pre-trained model in the designed GRU_δ to utilize both local and sequential features to minimize hedge risk. Specifically, the output of the proposed GRU_δ is a linear combination of the candidate output $\widehat{\delta}^M_{t,T,K}$ and the BS delta $\delta^{BS}_{t,T,K}$ computed from the implied volatility at the hedging time t for the option with expiry T and strike K. In addition, different combination formula are used when training for different types of options. For hedging a call option, the final output from GRU_δ is :

$$\delta^M_{t,T,K} = \widehat{\delta}^M_{t,T,K} \times W_\delta + \delta^{BS}_{t,T,K} \times (1 - W_\delta) \tag{4.1.2}$$

For hedging a put option, the final output from the model is:

$$\delta^M_{t,T,K} = -\widehat{\delta}^M_{t,T,K} \times W_\delta + \delta^{BS}_{t,T,K} \times (1 - W_\delta) \tag{4.1.3}$$

where $\widehat{\delta}^M_{t,T,K}$ is the candidate output. When $W_\delta = 0$, $\delta^{BS}_{t,T,K}$ is the output and, when $W_\delta = 1$, the output is the candidate $\widehat{\delta}^M_{t,T,K}$. The combination weight W_δ is defined in (4.1.1). The sigmoid function is used as the final activation function because, under the Black-Scholes-Merton framework [20], the range of the hedging position for call options is $[0,1]$ and the range of the hedging position for put options is $[-1,0]$. It can be easily shown that $\delta^M_{t,T,K}$, given by (4.1.2), is within

$[0,1]$ and $\delta^M_{t,T,K}$, given by (4.1.3), is within $[-1,0]$.

4.1.5 Robust Loss Function

At the hedging time t_i, for the data instance i with strike K_i, and expiry T_i , the local discrete hedging loss is:

$$loss_i = \Delta V^{mkt}_{t_i,T_i,K_i} - \Delta S_{t_i}\delta^M_{t_i,T_i,K_i}$$

where $\delta^M_{t_i,T_i,K_i}$ is the corresponding final output from GRU_δ, ΔS_{t_i} denotes the change in the market underlying price, and $\Delta V^{mkt}_{t_i,T_i,K_i}$ denotes the change in the market option price, see (2.1.1). Let M be the number of data instances. The objective function used in the data-driven regularized kernel hedging model in [143] is the mean squared loss,

$$MSE = \frac{1}{2M}\sum_{i=1}^{M} loss_i^2. \tag{4.1.4}$$

Since the quadratic error function is sensitive to outliers, in this work, we additionally incorporate the Huber loss function [110] below in the proposed GRU_δ:

$$HE = \frac{1}{M}\sum_{i=1}^{M} Huber(loss_i),$$

where $Huber(\cdot)$ is as defined below:

$$Huber(loss, \mathscr{T}) = \begin{cases} \frac{1}{2}loss^2, & \text{if } |loss| \le \mathscr{T} \\ \mathscr{T}(|loss| - \frac{1}{2}\mathscr{T}), & \text{otherwise} \end{cases}$$

The Huber loss [110] has been shown to be more robust than the squared loss with respect to outliers, when the threshold parameter $\mathscr{T}$ is carefully chosen *a priori*. In the context of machine learning, the parameter $\mathscr{T}$ can be tuned but this can be computationally expensive. In the proposed GRU_δ, since delta from the BS model with the implied volatility is used as a pre-trained model, we adaptively set the threshold parameter $\mathscr{T}$ to be absolute value of the hedging error from the Black–Scholes delta of the data instance i, i.e.,

$$\mathscr{T}_i = |\Delta V^{mkt}_{t_i,T_i,K_i} - \Delta S_{t_i}\delta^{BS}_{t_i,T_i,K_i}| \tag{4.1.5}$$

where $\delta^{BS}_{t_i,T_i,K_i}$ is the Black–Scholes delta using implied volatility for data instance i. The modified Huber loss becomes

$$MHuber(loss_i, \mathscr{T}_i) = \begin{cases} \frac{1}{2}loss_i^2, & \text{if } |loss_i| \le \mathscr{T}_i \\ \mathscr{T}_i(|loss_i| - \frac{1}{2}\mathscr{T}_i), & \text{otherwise} \end{cases}$$

Using the Huber loss with the adaptive threshholding parameter in (4.1.5), the objective for the proposed GRU_δ is:

$$MHE = \frac{1}{M}\sum_{i=1}^{M} MHuber(loss_i, \mathscr{T}_i) \tag{4.1.6}$$

4.2 Training GRU_δ

Next we discuss training GRU_δ, including initialization, pre-training, optimization, and regularization.

4.2.1 Initialization

When training a RNN model, an initial weight matrix is typically chosen as a random orthogonal matrix. Since the orthogonal initialization can often speed up training [124], all the weight matrices are initialized as orthogonal random matrices in training GRU_δ. The random orthogonal matrices are Householder transformations of random matrices [124].

4.2.2 Optimization

First-order optimization methods, such as stochastic gradient descent (SGD) and its extensions, are the most widely used optimization methods in machine learning, due to their low computational costs. Despite their wide usage, well-known deficiencies when training highly non-convex objective functions include relatively-slow convergence, sensitivity to hyper-parameter values (e.g., learning rate), stagnation at high training errors, and difficulty in escaping flat regions and saddle points [185].

More recently, improved SGD methods such as ADAM [120], have been proposed. These methods seem to achieve significantly better solutions compared with earlier SGD. However, it has recently been empirically observed in [155] that these algorithms sometimes fail to converge to a first-order critical point due to the exponential moving average used in these gradient descent algorithms.

Since the data size and the number of the parameters in GRU_δ are relatively small (which is a distinguishing characteristics of many financial data mining problems), it is computationally feasible to apply a second order optimization method. Furthermore, the recently proposed trust-region sub-problem solver [125] computes the trust region sub-problem solution iteratively requiring only Hessian-vector products. Consequently we use a trust region method, which is shown in Algorithm 1, when training GRU_δ. The meta-parameters for trust-region algorithm are shown in Table 4.1.

Algorithm 1: Trust-region Algorithm

Input: $\theta_0 \in \mathbb{R}^n$: initial vector of parameters.
$Obj(\theta)$: the objective function
$\mathscr{R}_0$: initial trust region radius
ε_θ: tolerance for the norm of the gradient
ε_r: tolerance for the trust region radius
η_{r_1}: first threshold for update the trust region radius
η_{r_2}: second threshold for update the trust region radius
$\gamma_u > 1$: ratio to increase the trust-region radius
$0 < \gamma_d < 1$: ratio to decrease the trust-region radius

Output: θ^*: the vector of parameters that minimize the objective function $Obj(\theta)$

begin
  $k = 0$;
  while $\|\nabla Obj(\theta_k)\|_2 \geq \varepsilon_\theta$ *and* $\mathscr{R}_k \geq \varepsilon_r$ **do**
    solve the trust-region subproblem [125]:

$$s_k^* = \underset{s}{\operatorname{argmin}}\, m_k(s) = Obj(\theta_k) + s^T \nabla Obj(\theta_k) + s^T \nabla^2 Obj(\theta_k) s$$
$$\text{sub to} \|s\|_2 \leq \mathscr{R}_k$$

    define

$$\mathscr{P}_k = \frac{Obj(\theta_k) - Obj(\theta_k + s_k^*)}{Obj(\theta_k) - m_k(s_k^*)}$$

    if $\mathscr{P}_k \geq \eta_{r_1}$ **then**
      $\theta_{k+1} = \theta_k + s_k^*$
    else
      $\theta_{k+1} = \theta_k$
    end
    Update the radius:

$$\mathscr{R}_{k+1} = \begin{cases} \gamma_u \|s_k^*\|_2, & \text{if } \mathscr{P}_k < \eta_{r_1} \\ \mathscr{R}_k, & \text{if } \eta_{r_1} \leq \mathscr{P}_k \leq \eta_{r_2} \\ \max(\gamma_u \|s_k^*\|_2, \mathscr{R}_k), & \text{if } \mathscr{P}_k > \eta_{r_2} \end{cases}$$

    k=k+1
  end

$$\theta^* = \theta_k$$

end

$\mathscr{R}_0$	ε_θ	ε_r	η_{r_1}	η_{r_2}	γ_u	γ_d
1.0	10^{-5}	10^{-8}	0.25	0.75	2	0.5

Table 4.1: Meta-parameters for the trust-region algorithm. The trust-region algorithm is used for training all the data-driven models based on neural network framework in this thesis.

4.2.3 Pre-training and Regularization

We use early stopping as the regularization [153]. At each training date, we reserve a fraction of the observed data to be the validation set. The rest of the observed data is used as the training set. The performance on the validation set is used to determine when over-fitting begins. Let θ denote the set of parameters to be learned for the proposed GRU_δ. After each training step k, the model performance is evaluated on the validation set and the associated parameters θ_k is recorded. Model training optimization is performed on the training set until the trust-region optimizer terminates. The parameters θ_k that achieve the best performance on the validation set is used to predict the hedging position for the testing data instances.

For GRU_δ, the training optimization problem is nonconvex, which can benefit from a good initial model. We choose the initial model for the training problem by matching the output of GRU_δ to the pre-trained BS model $\delta^{BS}_{t,T,K}$. Specifically, we determine the initial model for training optimization by solving the nonlinear least square problem below,

$$\min \frac{1}{2M} \sum_{i=1}^{M} (\delta^{M}_{t_i,T_i,K_i} - \delta^{BS}_{t_i,T_i,K_i})^2, \tag{4.2.1}$$

where the starting point for (4.2.1) is a set of randomly initialized weight matrices, as described in section 4.2.1.

Using a solution to (4.2.1) as the initial model, we train GRU_δ with the robust objective function (4.1.6). Recall that the output of GRU_δ, either (4.1.2) or (4.1.3), is a linear combination of the candidate output $\widehat{\delta}^{M}_{t,T,K}$ and the BS delta $\delta^{BS}_{t,T,K}$ computed from the implied volatility. When W_δ approaches 0, the output of GRU_δ converges to $\delta^{BS}_{t,T,K}$ as the output. When W_δ approaches 1, GRU_δ outputs candidate $\widehat{\delta}^{M}_{t,T,K}$. This simple combination structure allows the pre-training phase to be finished in just a few training steps.[1] The pre-training stage guarantees that the initial performance of the model on the validation set is close to $\delta^{BS}_{t,T,K}$. If the output from the model after training performs worse than $\delta^{BS}_{t,T,K}$ on the validation set, we use the initial model after pre-training.

Additionally the validation set is also used to select whether to use the mean square objec-

[1] Note that we will also stop the pre-training when $W_\delta > 0.97.5$ or $W_\delta < 0.025$. This is because, when we have $W_\delta > 0.97.5$ or $W_\delta < 0.025$, due to saturating gradient problem of the sigmoid function, the initial gradient, when training with actual local hedging objective, will be very small. This will slow down the first few learning steps when training with actual local hedging objective.

tive function (4.1.4) or the modified Huber loss objective (4.1.6) . The model trained with the objective function that performs better on the validation set is used when predicting the hedging position of the testing data instances, which are unobserved at the time of training.

4.2.4 Model Re-use and Re-initialization

In contrast to the regularized kernel method in [143], one advantage of the proposed GRU_δ is that the parameters from one trained model can be readily reused as the starting point for the model training on a subsequent rebalancing day. This allows a model to be updated more frequently, adapting to market changes without completely rebuilding a model. The kernel method in [143], on the other hand, recomputes the kernel matrix at each rebalancing time, which requires $O(m^3)$ computation assuming a completely new kernel matrix. Thus, updating the model frequently using the kernel method in [143] is computationally prohibitive.

An artificial neural network is known to potentially suffer catastrophic interference or catastrophic forgetting [136], i.e., a model forgets completely and abruptly previously learned information upon observing new information. When this happens, the resulting broken model usually has poor generalization ability. If a broken model with poor generalization ability is allowed to be continually updated, the performance of all the future models may be negatively affected. Thus, whenever a worse performance, in comparison to $\delta^{BS}_{t,T,K}$, is observed on a validation set, we re-initialize the parameters of GRU_δ and pre-train the model again. This avoids continually updating the broken model. If after model re-initialization and training, the performance of the proposed GRU_δ remains worse than that of $\delta^{BS}_{t,T,K}$ on the validation set, we simply output $\delta^{BS}_{t,T,K}$.

In summary, we initialize the model as described in section 4.2.1 and 4.2.3 on the first testing date. When we train the model, we reuse the parameters from the previously trained model as the starting point unless the performance of the proposed GRU_δ is worse than that of $\delta^{BS}_{t,T,K}$, in which case we re-initialize the parameters of the proposed GRU_δ and train the model again.

4.3 Alternative Data-Driven Hedging Models Under Neural Network Framework

Lastly, before jumping into the real data experimental results for local discrete hedging in Chapter 5, we introduce the following supplementary models:

1. To gain insights on the roles of the decoder and encoder in the proposed GRU_δ, we also consider a decoder only model NN_δ, which corresponds to removing the encoder from GRU_δ in Figure 4.3. We train NN_δ in the same way as described in section 4.2.

2. To gain insights on the role of output gate and robust Huber loss, we remove the output gate part of the decoder model. The resulting model is shown in Figure 4.4. In other words,

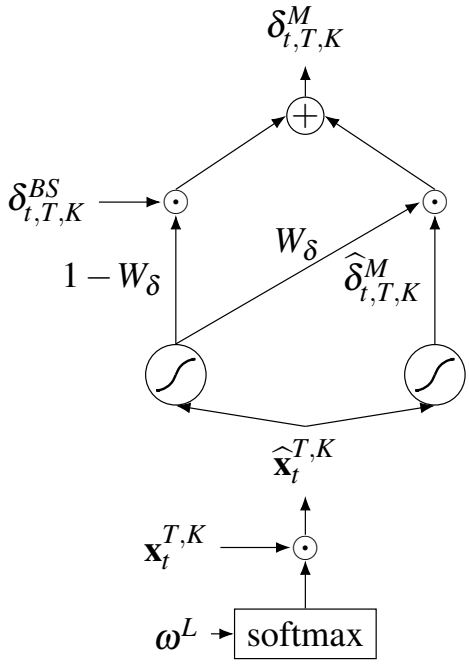


Figure 4.3: NN_δ: decoder GRU only. This simplified model only retains the decoder. The decoder computes the hedging position solely based on the information vector $\mathbf{x}^{T,K}_t$ observed at time t. A candidate output $\widehat{\delta}^M_{t,T,K}$ is produced. The final output $\delta^M_{t,T,K}$ is computed based on the linear combination of BS delta $\delta^{BS}_{t,T,K}$ and the candidate output $\widehat{\delta}^M_{t,T,K}$. The combination weight is determined by W_δ. The feature weight ω^L is used to produce the weighted local feature $\widehat{\mathbf{x}}^{T,K}_t$. The weighting acts as a feature selection process. Each edge in the graph has an arrow on it, pointing from a node whose output is used by the node pointed by the arrow as an input.

the output δ^M is purely based on $\widehat{\mathbf{h}}_E$ and $\widehat{\mathbf{x}}^{T,K}_t$:

$$\delta^M_{t,T,K} = sigmoid(\mathbf{v}^T_{out}\ tanh(\boldsymbol{U}_{out}\widehat{\mathbf{h}}_E + \boldsymbol{W}_{out}\widehat{\mathbf{x}}^{T,K}_t + \mathbf{b}_{out})).$$

We denote the model shown in Figure 4.4 as GRU_c. The training procedure of GRU_c is slightly different from GRU_δ: the objective function is fixed to be the mean squared loss. Therefore, by comparing the hedging performance from GRU_c and GRU_δ, we can see the importance of the robust huber losses and the output gate mechanism.

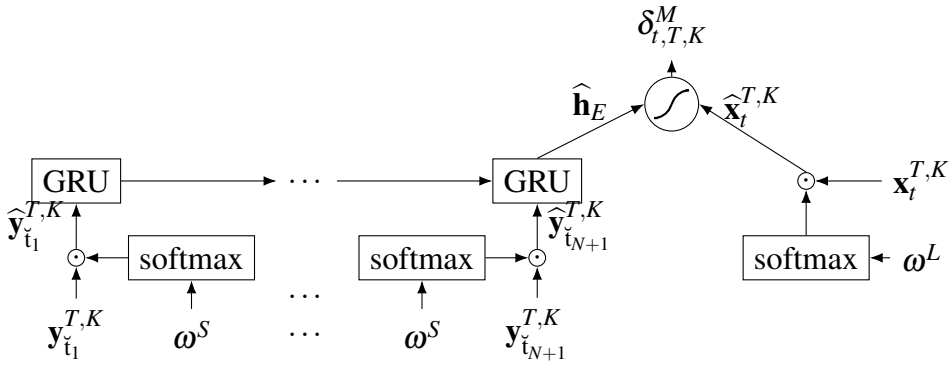


Figure 4.4: GRU$_c$: This simplified model removes the output gate in the decoder. The encoder summarizes the time series $\mathbf{Y}^{T,K}_t = \left[\mathbf{y}^{T,K}_{\check{t}_1}, \ldots, \mathbf{y}^{T,K}_{\check{t}_{N+1}}\right]$ as a succinct vector $\widehat{\mathbf{h}}_E$. The decoder outputs the hedging position based on the vector $\widehat{\mathbf{h}}_E$ and the local feature vector $\mathbf{x}^{T,K}_t$ observed at the hedging time t directly. The output gate is removed and we do not combine the output with $\delta^{BS}_{t,T,K}$ using linear combination as in Figure 4.1. Each edge in the graph has an arrow on it, pointing from a node whose output is used by the node pointed by the arrow as an input.

Chapter 5

Local Discrete Hedging Performance Comparison Using S&P 500 index Options

Using the S&P 500 (European) index option market data from January 2, 2001 to August 31, 2015, we compare hedging performance of different hedging strategies.

Following [112], we use the Gain ratio below as the evaluating measure for the hedging performance:

$$\text{GAIN} = 1 - \frac{\sum_{i=1}^{m} \left((\Delta V^{mkt}_{t_i,T_i,K_i} - \delta_{t_i,T_i,K_i} \, \Delta S_{t_i} \right)^2}{\sum_{i=1}^{m} \left((\Delta V^{mkt}_{t_i,T_i,K_i} - \delta^{BS}_{t_i,T_i,K_i} \, \Delta S_{t_i} \right)^2},$$

where m is the total number of data instances to be evaluated. In addition, for the data instance i, corresponding to hedging option with expiry T_i and strike K_i at hedging time t_i, we recall that $\delta^{BS}_{t_i,T_i,K_i}$ is the implied Black–Scholes delta, δ_{t_i,T_i,K_i} is the hedging position from the method under consideration, e.g., SABR delta, MV delta, and output from DKL_{SPL} or GRU_{δ}. In addition $\Delta V^{mkt}_{t_i,T_i,K_i}$ is the change in option price, and ΔS_{t_i} is the change in the underlying price which are given by (2.1.1).

5.1 Data and Experimental Setting

The option data used in this study comes from the OptionMetric database and the data is processed following the same procedure described in [112]. On each day, the mid-price from the bid and ask is used as the price of the option on that day. The closing price for the underlying is regarded as the daily underlying price. Options with time-to-expiry less than 14 days are removed from the data set. The call options with the BS delta $\delta^{BS}_{t,T,K}$ from the implied volatility outside of [0.05, 0.95] and the put options with the BS delta $\delta^{BS}_{t,T,K}$ from implied volatility outside of [-0.95, -0.05] are also removed from the data set. These removed option prices are noisy and unreliable

since they correspond to either deeply out-of-money options, deeply in-money option, or very short term options. As in [112], the option data instances are divided into nine buckets according to the Black–Scholes delta δ^{BS} from the implied volatility (rounded to the nearest tenth) for detailed hedging performance comparisons.

Inputs to GRU_δ, $\mathbf{Y}_t^{T,K} \in \mathbb{R}^{d_s \times (N+1)}$, are 6 sequential features and the length of each sequence is 6, i.e., $d_s = 6$ and $N = 5$. At each hedging time t, the $\mathbf{Y}_t^{T,K}$ is the matrix recording the historical time series of the feature listed in Table 5.1.

option mid price
option implied volatility
option BS delta
option BS gamma
option BS vega
moneyness

Table 5.1: Sequential features for model GRU_δ at time t are the historical time series of the features listed in this table

At each hedging time t, there are 10 local features $\mathbf{x}_t^{T,K}$, which is given in Table 5.2 :

moneyness
time to expiry
index close price
option bid price
option offer price
option BS delta
δ_{MV} from equation (2.3.5)
option implied volatility
option BS gamma
option BS vega

Table 5.2: Local features $\mathbf{x}_t^{T,K}$ at time t for model GRU_δ

The number of hidden states, for the single-layer GRU encoder, the neural network outputting $\widehat{\delta}^M_{t,T,K}$, and the neural network outputting W_δ in Figure 4.1, are all set to be 5. In comparison with most RNN applications, we train a relatively small model and the number of data instances available for the hedging problem is also relatively small. We note that our training and testing data instances are generated from daily observations with a moving window. On each day, data instances observed in the previous 36 months are reserved as the training set and the validation set. For daily and weekly hedging, the validation consists of the data instances observed during the past 30 days. For monthly hedging, the validation set is the data instances observed during the previous 60 days. This is because, for monthly hedging, the ΔS_t and $\Delta V^{mkt}_{t,T,K}$ of many data

instances during the last 30 days have not been observed.[1] The data instances, which are observed in the previous 36 months but are not used in the validation set, are used as the training set. The models are updated daily.

We note that the length of our backtest time is longer than a decade. Since learning a single $\mathrm{DKL}_{\mathrm{SPL}}$ model for all options will be computationally expensive, a separate model is built for each delta bucket when training $\mathrm{DKL}_{\mathrm{SPL}}$, which has the feature set # 2 below

$$\{\text{moneyness}(\frac{S}{K}), \text{time-to-expiry}(T-t), \text{Black–Scholes delta}(\delta^{BS}_{t,T,K})\}.$$

In addition, we use the efficient leave-one-out cross-validation (LOOCV) as in section A to choose regularization parameter λ_P for $\mathrm{DKL}_{\mathrm{SPL}}$. For each month, LOOCV is used to choose a value of the regularization parameter from a candidate set:

$$\lambda_P \in \{10^7, 10^6, 10^5, 10^4, 10^3, 10^2, 10^0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}.$$

For the call option, the $\mathrm{DKL}_{\mathrm{SPL}}$ for each bucket is estimated using all options **traded** in a 36 months window. For the put option, there is much larger variation in the number of data instances in each bucket. To ensure a reasonable training data size, we have used a time window of 24 months for delta buckets -0.1 and -0.2, window of 36 months for delta buckets -0.3 and -0.4, window of 48 months for delta buckets -0.5 and -0.6, and window of 72 months for delta buckets -0.7, -0.8 and -0.9. Hence, the training data for puts ranges from Jan 2001 to August 31 2015. We train the direct hedging function model $\mathrm{DKL}_{\mathrm{SPL}}$ using all options **traded** in a fixed time window, as described above, and determine the hedge each day in the following month using the trained model, following the same sliding training-testing window procedure in [112]. Furthermore, the $\mathrm{DKL}_{\mathrm{SPL}}$ model is updated monthly due to the limitation of computationally cost.

The training set, the validation set, and the testing set are the same for GRU_δ, GRU_c, and NN_δ. The training procedure is the same for GRU_δ and NN_δ which is described in section 4.2. In addition, GRU_c is trained using the data instances, observed in the previous 36 months and procedure described as in section 4.3.

For additional reference, we also include the performance of the MV model (2.3.5) and Bartlett delta on weekly and monthly hedging. For MV in Table 5.3 & 5.4, on each day, the model parameter, a, b and c are estimated using all traded options in a 36-month-window. [2]

For $\mathrm{DKL}_{\mathrm{SPL}}$ and GRU_δ, we present out-of-sample daily hedging performance test result using either traded data or all data in the database.

[1] As an example, assume the model is trained on 2022-01-03. The previous 30 days period as of 2022-01-03 is from 2021-12-03 to 2022-01-02. However, we cannot observe ΔS_t and $\Delta V^{mkt}_{t,T,K}$ for any of those dates since 20 business days away from 2021-12-03, which is the starting point of this period, is 2022-01-05, which is after the as of date:2022-01-03.

[2] Note that, in the OptionMetric Database, not all the option data instances are actually traded. For many data instances, the trading volumes are zero. It is not clear to us whether the reported test results in [112] are computed using all data or traded data only.

5.1.1 Weekly and Monthly Hedging Comparisons

We first present hedging comparisons for weekly and monthly hedging. We assume a period of 5-business-days for weekly hedging and a period of 20-business-days for monthly hedging. We note that the MV hedging method is only considered for daily hedging in [112].

Specifically, we compare the following methods,

- GRU_δ: The sequential learning framework trained as in section 4.2.
- MV: MV hedging based on formulation (2.3.5),
- Bartlett: Barlett corrective delta based on (2.3.18),
- DKL_{SPL}: Direct spline kernel hedging method as in 3.2,
- NN_δ: NN with the decoder only described in Figure 4.3.
- GRU_c: The sequential learning model excluding the output gate and trained with MSE objective only.

Table 5.3 present comparisons for calls while Table 5.4 demonstrate comparisons for puts. We further note that the parameters of MV are updated daily in this section while the parameters of MV in [112] are updated monthly. We also note that the out-of-sample results for the MV model in [112] are obtained with daily changes of the index price and option price in [112] while, in Table 5.3 and 5.4, the out-of-sample results are obtained with weekly and monthly changes of the index price and option price. The SABR model used to compute the Bartlett delta is also calibrated daily.

From Table 5.3 and Table 5.4, we observe that, overall, GRU_δ outperforms GRU_c, MV, Bartlett, DKL_{SPL}, and NN_δ. In addition, the hedging performance of the four data-driven models, DKL_{SPL}, NN_δ, GRU_c, and GRU_δ, is significantly better than that of the MV model and Bartlett except for put option weekly hedging where Bartlett correction performs better than DKL_{SPL}. This is not surprising since the MV formula (2.3.5) and Bartlett formula (2.3.18) are based on instantaneous hedging analysis. Furthermore, by comparing DKL_{SPL} with NN_δ, we see that the decoder only NN_δ still achieves significant improvement over DKL_{SPL}. The improvement possibly comes from inclusions of more local features in NN_δ and more frequent updates of NN_δ, timely addressing the market changes. The overall improvement of GRU_δ over NN_δ illustrates the important role of the GRU encoder, which incorporates sequential feature information, in weekly and monthly hedging. Lastly, the overall improvement of GRU_δ over GRU_c illustrates the important role of the output gate, the robust huber loss and the robust training procedure in section 4.2.

Delta	Comparing Model(%)											
	Weekly						Monthly					
	MV	Bartlett	DKL_{SPL}	NN_δ	GRU_c	GRU_δ	MV	Bartlett	DKL_{SPL}	NN_δ	GRU_c	GRU_δ
0.1	26.3	-16.9	38.9	35.6	36.6	**47.8**	13.5	-8. 2	22.7	29.7	34.8	**53.9**
0.2	21.6	-5.6	29.0	36.4	39.6	**48.5**	16.4	0.4	23.5	38.4	38.9	**51.7**
0.3	20.1	11.9	23.5	38.6	39.7	**48.5**	17.9	2.1	24.0	40.2	41.7	**50.2**
0.4	18.1	17.3	20.8	38.7	38.9	**45.9**	16.9	2.7	21.0	38.6	42.6	**47.8**
0.5	16.0	21.7	19.9	42.3	37.5	**46.6**	15.2	5.7	13.5	36.3	42.3	**44.5**
0.6	12.1	24.1	17.3	43.4	33.5	**44.8**	12.7	8.4	14.3	36.0	40.7	**44.6**
0.7	8.1	26.3	16.8	**45.6**	31.1	43.9	5.9	7.5	6.1	30.2	26.3	**35.3**
0.8	3.7	25.5	12.5	39.6	31.7	**37.7**	-1.2	4.2	5.3	22.3	26.3	**24.8**
0.9	2.4	21.7	6.2	26.3	**28.7**	16.4	-1.8	9.8	4.1	**21.1**	17.3	10.5
Overall	15.1	18.6	20.2	39.9	33.5	**43.7**	13.4	4.5	16.3	35.4	38.0	**44.5**

Table 5.3: S&P 500 call options hedging comparison on traded data, bold entries indicating best Gain. The Gain ratio is a measure for the local hedging performance. The larger the gain ratio is, the better improvement the model achieves over the baseline BS delta hedging method in terms of local hedging risk. The gain ratio is reported on different delta buckets.

Delta	Comparing Model(%)											
	Weekly						Monthly					
	MV	Bartlett	DKL_{SPL}	NN_δ	GRU_c	GRU_δ	MV	Bartlett	DKL_{SPL}	NN_δ	GRU_c	GRU_δ
-0.9	23.9	9.1	10.1	29.5	32.1	**34.7**	16.9	1.2	6.5	28.3	27.4	**32.6**
-0.8	21.5	-0.1	18.3	39.6	40.1	**44.2**	11.5	5.6	6.1	41.7	35.6	**49.5**
-0.7	19.1	0.4	20.2	44.0	39.6	**49.6**	9.6	6.7	7.3	43.4	41.1	**52.4**
-0.6	16.1	9.1	20.8	43.0	40.3	**51.3**	8.1	8.6	10.3	42.1	41.5	**51.6**
-0.5	15.3	20.9	22.4	43.4	36.3	**53.5**	7.7	13.2	13.9	41.2	42.7	**51.4**
-0.4	12.3	25.7	21.0	41.4	34.6	**53.2**	6.8	14.4	15.6	40.7	42.9	**53.4**
-0.3	9.7	29.4	22.2	38.2	37.4	**51.1**	4.7	13.6	19.5	34.1	42.5	**48.4**
-0.2	7.8	33.1	20.8	29.8	25.4	**46.3**	2.9	10.7	20.6	21.7	24.7	**44.7**
-0.1	4.9	30.5	19.2	15.5	10.4	**37.2**	-1.8	10.8	13.0	12.3	15.1	**26.8**
Overall	14.4	26.4	20.4	38.7	32.5	**49.1**	8.6	12.1	13.5	38.6	40.5	**49.5**

Table 5.4: S&P 500 put options hedging comparison on traded data, bold entries indicating best Gain. The Gain ratio is a measure for the local hedging performance. The larger the gain ratio is, the better improvement the model achieves over the baseline BS delta hedging method in terms of local hedging risk. The gain ratio is reported on different delta buckets.

5.1.2 Feature Importance

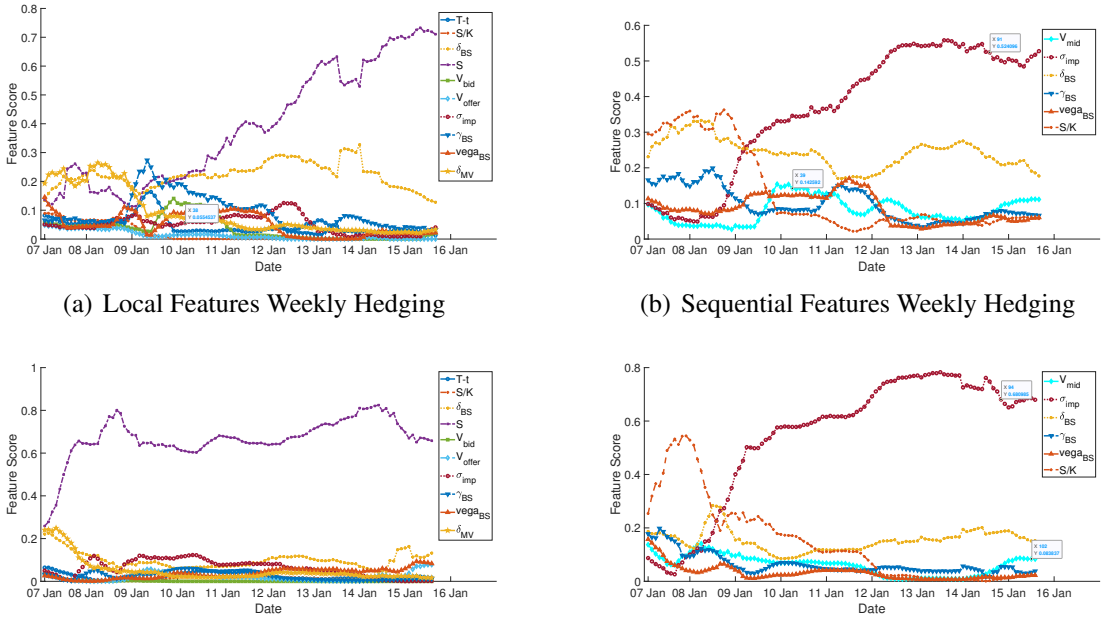
Next, we discuss feature importance in the trained model GRU_δ. Specifically, we show how the normalized feature weight, for both the local features $\mathbf{x}_t^{T,K}$ and for the sequential features $\mathbf{X}$, changes from Jan 2007 to Aug 2015. The normalized feature weight vector are:

$$\frac{exp(\omega^L)}{\sum_{i=1}^{d_l} exp(\omega_i^L)}$$

and

$$\frac{exp(\omega^S)}{\sum_{i=1}^{d_s} exp(\omega_i^S)}$$

respectively. The feature score, shown in Figure 5.1& 5.2, represents the normalized feature weights averaged over a calendar month for the model trained using traded data instance. Note that the model is updated daily and the normalized feature weights vary with the training date.



(a) Local Features Weekly Hedging

(b) Sequential Features Weekly Hedging

(c) Local Features Monthly Hedging

(d) Sequential Features Monthly Hedging

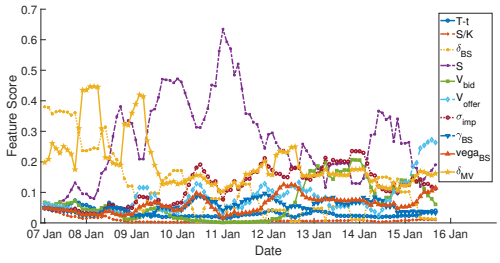
Figure 5.1: Feature scores for weekly and monthly hedging models GRU_δ on traded S&P500 call option data.

From subplots (a) and (b) in Figure 5.1, it can be observed that, for weekly hedging call options, the index price S is ranked as the most important local feature after Jan 2010. Moreover, the average weight for the index price S exhibits an increasing trend after Jan 2009. The past implied volatility sequence is ranked as the most important sequential feature after March 2009. The average weight for the past implied volatility sequence also exhibits an increasing trend after

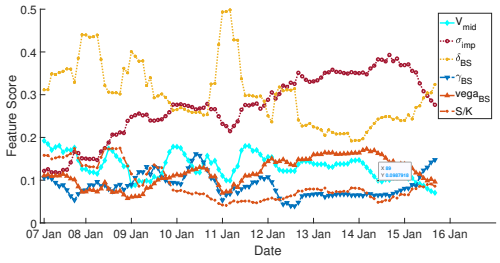
Jan 2009. From subplots (c) and (d) in Figure 5.1, it can be observed that, for monthly hedging the call option, the index price S is always ranked as the most important local feature. In addition, the average weight of the index price S is always higher than 0.5 after Jan 2008. The past implied volatility sequence is always ranked as the most important sequential feature after Jan 2009. The average weight for the past implied volatility sequence also is always greater than 0.4 after Jan 2008.

For the put option, the situation is slightly different. From subplots (a) and (b) in Figure 5.2, we can see that, for weekly hedging the put option, the index price S is often ranked as the most important local feature. The past implied volatility sequence and the past BS delta sequence are identified as the two most important sequential features. From subplots (a) and (b) in Figure 5.2, we can see that, for monthly hedging put options, the index price S is ranked as the most important local feature after Jan 2008. The past implied volatility sequence is ranked as the most important sequential feature after Jan 2013.

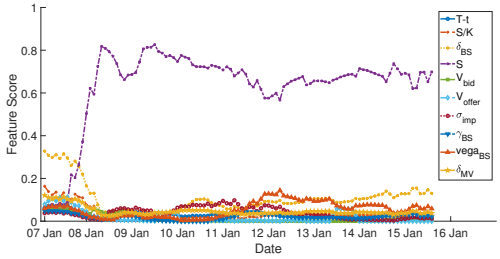
Overall, the index price S has often been identified as an important local feature for hedging both call and put options. Since the BS delta of the implied volatility is one of the local features, this confirms the fact that the BS delta does not capture all dependence on the underlying In addition, the past implied volatility has often been identified as an important sequential feature for hedging both calls and puts.



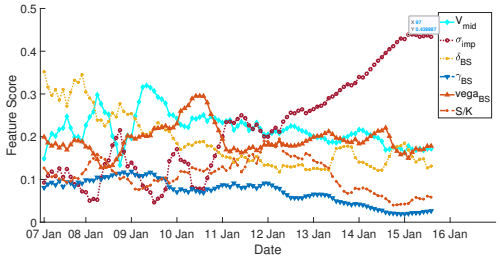
(a) Local Features Weekly Hedging



(b) Sequential Features Weekly Hedging



(c) Local Features Monthly Hedging



(d) Sequential Features Monthly Hedging

Figure 5.2: Feature score for weekly and monthly hedging models GRU_{δ} on traded S&P500 put option data

5.1.3 Daily Hedging Comparison

In section 5.1.1 - 5.1.2, we have presented hedging comparisons for weekly and monthly hedging. Now we compare the proposed GRU_δ with the following methods for daily hedging,

- MV: MV hedging based on formulation (2.3.5),
- LVF: MV hedging in [112] from local volatility function based on formulation (2.3.14),
- SABR_{MV} : MV hedging in [112] from SABR model based on formulation (2.3.20),
- δ^{BS}: implied volatility Black–Scholes delta,
- Bartlett: Bartlett formula based on (2.3.18),
- DKL_{SPL}: Direct spline kernel hedging method as in 3.2.

We have omitted NN_δ and GRU_c in the daily hedging comparison since it has been shown to underperform GRU_δ. But we have added LVF and SABR_{MV} since they are based on instantaneous hedging analysis and are more suitable for daily hedging. Overall, all methods perform similarly in terms of daily hedging with data-driven model slightly outperforming the parametric models.

Since the same S&P 500 index option data from the OptionMetric Database is used here, we simply quote the results presented in [112] for MV, LVF, and SABR_{MV}, and the results presented in [143] for DKL_{SPL}.

As stated in [112] and section 2.3.1, the SABR_{MV} model is calibrated daily, from which the hedge position is determined and applied the next day. For LVF, the partial derivative of the expected implied volatility to the underlying is calculated from the slope of the observed implied volatility smile daily. For MV, the model parameter, a, b and c are estimated using all options in a 36-month-window and then applied to determine the hedging position daily in the subsequent month.

Table 5.5 presents daily hedging comparisons for call options. From Table 5.5, using either traded data or all data for testing, GRU_δ outperforms the minimum variance hedging methods reported in [112] in the overall performance. For most delta buckets, delta greater than or equal to 0.3, the performance of GRU_δ is better than those of the minimum variance hedging methods. However, for the bucket of the delta value 0.1 or 0.2, the performance is slightly worse.

In addition, from Table 5.5, the performance of the GRU_δ model is slightly better than that of the kernel hedging DKL_{SPL} in overall performance. For the the delta buckets 0.3-0.9, GRU_δ outperforms the direct kernel hedging DKL_{SPL}. However, GRU_δ performs slightly worse for the delta bucket 0.1 and 0.2.

Table 5.6 presents daily hedging comparisons for put options. Table 5.6 shows that both overall performance and the performances for different delta buckets of the GRU_δ model are

better than those from minimum variance hedging methods in [112] and the direct kernel hedging DKL_{SPL}, for testing result using all data. Similarly to the call, GRU_δ has model bold entries in most of delta buckets, indicating outperforming other methods.

Both GRU_δ and DKL_{SPL} are non-parametric data-driven hedging methods, with a main difference in features included. The hedging performance comparison between the two suggests that including the sequential features and more local features improves the overall hedging performance slightly for daily hedging.

Delta	MV (%)	SABR_{MV}(%)	LVF(%)	Bartlett		Data-Driven Model			
						DKL_{SPL} (%)		GRU_δ (%)	
				Traded	All	Traded	All	Traded	All
0.1	42.1	39.4	42.6	29.0	35.1	**47.1**	**48.6**	32.3	33.8
0.2	35.8	33.4	36.2	28.2	32.3	**37.8**	**40.0**	33.7	36.4
0.3	31.1	29.4	30.3	27.7	28.9	34.1	35.1	**34.1**	**35.5**
0.4	28.5	26.3	26.7	28.7	27.3	32.3	32.0	**33.7**	**34.2**
0.5	27.1	24.9	25.5	26.9	26.7	29.3	29.4	**35.1**	**33.0**
0.6	25.7	25.2	25.2	28.3	26.6	29.9	28.4	**35.6**	**32.1**
0.7	25.4	24.7	25.8	28.5	26.4	29.0	26.8	**31.8**	**29.7**
0.8	24.1	23.5	25.4	23.1	24.9	25.9	24.7	**28.6**	**26.5**
0.9	16.6	17.0	16.9	14.0	15.6	17.7	13.9	**19.3**	**18.9**
Overall	25.7	24.6	25.5	27.1	24.8	31.3	26.0	**32.9**	**28.7**

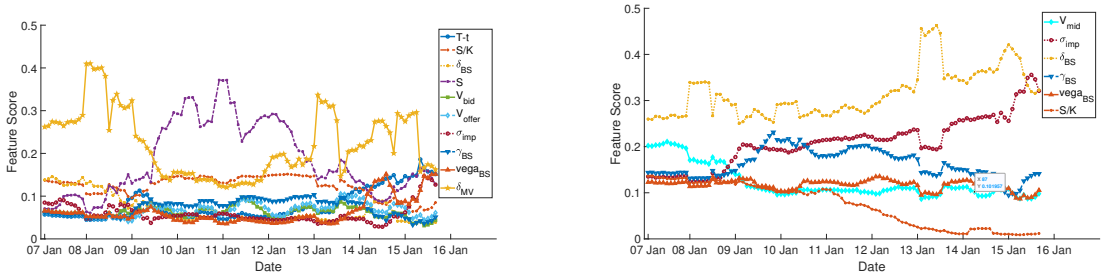
Table 5.5: S&P 500 call option hedging for 1-business day: bold entries indicating best Gain. The Gain ratio is a measure for the local hedging performance. The larger the gain ratio is, the better improvement the model achieves over the baseline BS delta hedging method in terms of local hedging risk. The gain ratio is reported on different delta buckets.

We similarly demonstrate feature importance in Figure 5.3&5.4 for daily hedging. From Figure 5.3, it can be observed that, for daily hedging call options, MV delta δ^{MV} and index price S are the local features that are often ranked as the most important under GRU_δ. The history of BS delta δ^{BS} is ranked as the most important sequential feature during most of the months. The past implied volatility sequence is often ranked as the second most important sequential feature.

From subplots (a) and (b) in Figure 5.4, it can be observed that the index price S is often ranked as the most important local feature for daily hedging put options. The past implied volatility sequence, the past BS delta sequence, and the past option price V_{mid} are often identified as the three important sequential features. A natural extension is to extend the sequential local hedging model to be a total risk hedging model where we rebalance multiple times until the expiries of the options. In the following chapter 6, we discuss the challenges associated with building such models and present our exploration on the data-driven total risk hedging model.

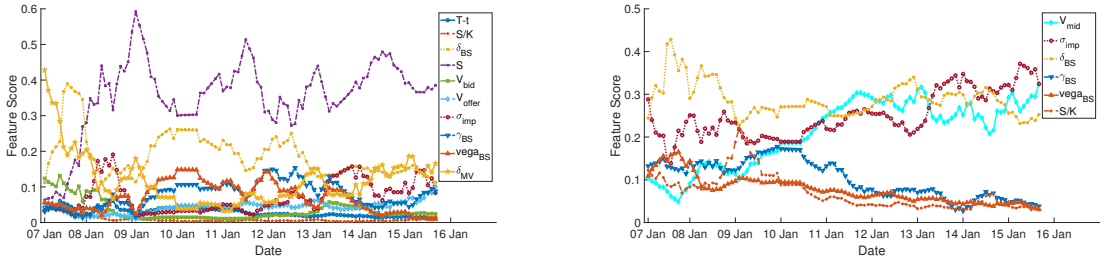
Delta	MV (%)	$SABR_{MV}$(%)	LVF(%)	Bartlett		Data-Driven Model			
				Traded	All	DKL_{SPL} (%)		GRU_{δ} (%)	
						Traded	All	Traded	All
-0.9	15.1	11.2	-7.4	9.1	23.4	8.6	13.6	**15.1**	**17.2**
-0.8	18.7	19.6	6.8	3.2	21.9	6.5	16.7	**23.2**	**28.5**
-0.7	20.3	17.7	9.1	1.5	20.1	10.6	19.8	**28.5**	**32.8**
-0.6	20.4	16.7	9.2	6.1	19.2	14.9	21.0	**28.3**	**33.9**
-0.5	22.1	16.7	10.8	15.5	21.3	22.5	23.1	**29.2**	**34.5**
-0.4	23.8	17.7	12.0	21.0	24.4	24.2	25.2	**29.9**	**34.7**
-0.4	27.1	21.7	16.8	26.7	29.0	27.7	28.3	**30.6**	**33.6**
-0.2	29.6	25.8	20.6	29.3	**31.6**	30.1	30.8	25.4	29.9
-0.1	27.5	26.9	17.7	**31.4**	**32.5**	29.1	31.2	18.7	21.4
Overall	22.5	19.0	10.2	20.0	24.8	23.4	23.2	**26.2**	**29.7**

Table 5.6: S&P 500 put option hedging for 1-business day: bold entries indicating best Gain. The Gain ratio is a measure for the local hedging performance. The larger the gain ratio is, the better improvement the model achieves over the baseline BS delta hedging method in terms of local hedging risk. The gain ratio is reported on different delta buckets.



(a) Local Features Daily Hedging (b) Sequential Features Daily Hedging

Figure 5.3: Feature score for daily hedging model GRU_{δ}: S&P500 call option (traded data)



(a) Local Features Daily Hedging (b) Sequential Features Daily Hedging

Figure 5.4: Feature score for daily hedging model GRU_{δ}: S&P500 put option (traded data)

Chapter 6

Data-Driven Sequential Learning for Total Hedging Risk

As we have discussed in section 2.1, in reality, one often want to hedge until the expiry of the option. Total risk measures the hedging error from a dynamic hedging strategy in the entire hedging horizon. While mininimizing local risk has the effect of limiting the total hedging error, we investigate here whether minimizing the total hedging risk directly can lead to better total risk mimization strategy. From the model perspective, such enhancement is achievable. Indeed, recently a deep hedging model was proposed to minimize the total option hedging risk evaluated at the option expiry [28]. It is shown that using a RNN to represent the hedging position model can be a computationally efficient framework to determine the optimal hedging function when the market is incomplete, e.g., under the transaction cost [28].

Although minimizing the total hedging risk, which is the hedging portfolio value at the expiry T, is more desirable, there are several major obstacles in obtaining enough market data to build a data-driven total hedging model due to the fact that options listed in the exchanges often have fixed expiry dates (e.g., once a month for S&P500 index options). The deep hedging model [28] is only built on synthetic data. Due to the lack of market data needed to build a model, applying the deep hedging model [28] can be challenging in real world applications. [1]

In this chapter, we provide a technique to deal with issue of lack of market data. In addition, an sequential total risk hedging model $\text{GRU}_{\text{TOTAL}}$ is introduced to minimize a discrete total risk hedging objective. We then build the total risk hedging model $\text{GRU}_{\text{TOTAL}}$ based on the data augmentation technique we propose and demonstrate its effectiveness and the performance of the total risk hedging model $\text{GRU}_{\text{TOTAL}}$ using real market data experiments. The goal in this chapter is to learn a hedging model for hedging option from a total hedging horizon of N_H days to expiry.

[1] Before 2016, the expiry date for S&P500 index options is fixed as the third Friday of each month. The CBOE introduces weekly S&P500 option in 2016 and daily S&P 500 options in 2022. However, in this thesis, the option data is gathered between 1996 and 2015, therefore, data augmentation is still needed. We comment that if we apply our data-driven model as of now, the lack of data issue will be significantly improved

6.1 Total Risk Hedging Model

In this section, we describe the total risk hedging model. Figure 6.1 depicts the proposed total risk GRU hedging model $\text{GRU}_{\text{TOTAL}}$, which is an extension from the local hedging model GRU_{δ} we proposed in Chapter 4. This model uses the sequential features, which encode information of two consecutive re-balancing time steps. In addition, the model uses the hedging position from the previous re-balancing time as the input. The output hedging position is used as the input for the next re-balancing time.

Following the discrete total risk definition in section 2.1.2, consider a hedging portfolio which is composed of:

- A short position on option $V_{t,T,K}$.
- $\delta^M_{t,T,K}$ shares of S_t.
- An amount in a risk-free bank account B_t.

As discussed in section 2.1.2, the final hedging portfolio value at T is:

$$\text{Risk}^{total}_{t_0,T,K} = \sum_{j=0}^{N_{rb}-1} \left\{ \left[\frac{S_{t_{j+1}}}{D(t_{j+1},T)} - \frac{S_{t_j}}{D(t_j,T)} \right] \delta^M_{t_j,T,K} \right\} + \frac{V_{t_0,T,K}}{D(t_0,T)} - V_{T,T,K} \tag{6.1.1}$$

where $D(t,T) = e^{-r(T-t)}$ is the discount factor and $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$ is the set of rebalacing time. Note that, for testing scenarios, following the scenario construction procedure as in Algorithm 7, we can guarantee that $V_{t_0,T,K}$, $V_{T,T,K}$, and S_t in equation (6.1.1) are all directly from market instead of model.

Now, consider at a rebalancing time $t \in \{t_0, t_1, \ldots, t_{N_{rb}-1}\}$, with a strike K, and an expiry T. Assume we have computed the hedging position at the previous rebalancing time $t - \Delta t$: $\delta_{t-\Delta t,T,K}$. Let Δt_d denote the time interval for sequential information recording. In the subsequent empirical study, the interval Δt_d equals one-day. Please note this setting is the same for local risk model described in Chapter 4. We denote the sequential features recording the daily history for hedging the option with expiry T and strike K as

$$\mathbf{Y}^{T,K}_t = \left[\mathbf{y}^{T,K}_{t-N\Delta t_d}, \ldots, \mathbf{y}^{T,K}_t \right]$$

For notational simplicity, we denote $\check{\mathrm{t}}_i = t - (N+1-i)\Delta t_d$ with $i = 1, \ldots, N+1$, we thus have:

$$\mathbf{Y}^{T,K}_t = \left[\mathbf{y}^{T,K}_{\check{\mathrm{t}}_1}, \ldots, \mathbf{y}^{T,K}_{\check{\mathrm{t}}_{N+1}} \right]$$

The vector $\mathbf{y}^{T,K}_{\check{\mathrm{t}}_i} \in \mathbb{R}^{d_s}$ has d_s features at time $\check{\mathrm{t}}_i$ in the input sequential feature where d_s is the dimension of the sequential feature $\mathbf{Y}^{T,K}_t$, and $N+1$ is the length of the sequential feature sequence.

We set $N = \frac{\Delta t}{\Delta t_d}$. Thus $\mathbf{Y}_t^{T,K}$ contains the sequential information in between two consecutive rebalancing time t and $t-\Delta t$. The encoder transforms information from the sequential feature $\mathbf{Y}_t^{T,K}$ to a fixed-sized vector $\widehat{\mathbf{h}}_E$ and the decoder makes the final prediction based on both $\widehat{\mathbf{h}}_E$ and the previous hedging position $\delta^M_{t-\Delta t,T,K}$. The overall structure of the proposed model is illustrated in Figure 6.1. Note that for the initial hedging date $t = t_0$, the previous hedging position is set as 0.

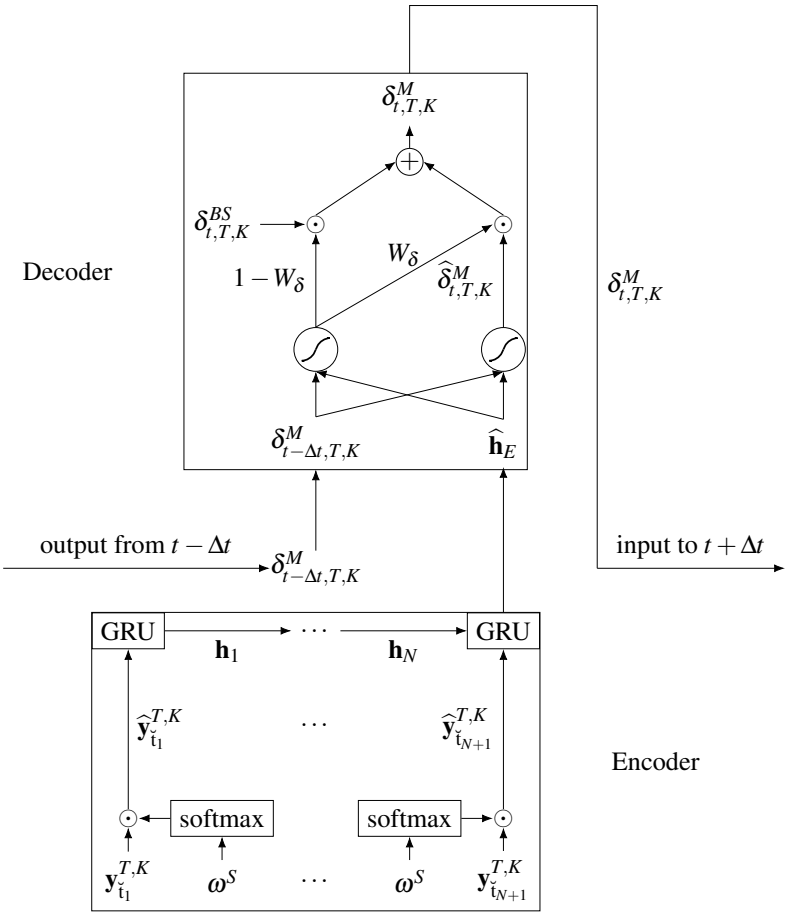


Figure 6.1: $\text{GRU}_{\text{TOTAL}}$: GRU encode-decoder total hedging model. The encoder summarizes the time series $\mathbf{Y}_t^{T,K} = \left[\mathbf{y}^{T,K}_{\check{\mathbf{t}}_1}, \ldots, \mathbf{y}^{T,K}_{\check{\mathbf{t}}_{N+1}}\right]$ as a succinct vector $\widehat{\mathbf{h}}_E$. The decoder outputs the hedging position based on the vector $\widehat{\mathbf{h}}_E$ and the previous hedging position $\delta^M_{t-\Delta t,T,K}$ observed at the hedging time t. More specifically, in the decoder, a candidate output $\widehat{\delta}^M_{t,T,K}$ is firstly produced. The final output $\delta^M_{t,T,K}$ is computed based on the linear combination of BS delta $\delta^{BS}_{t,T,K}$ and the candidate output $\widehat{\delta}^M_{t,T,K}$. The combination weight is determined by W_δ. The feature weight ω^L is used to compute weighted sequential feature $\widehat{\mathbf{y}}^{T,K}_{\check{\mathbf{t}}}$. The weighting acts as a feature selection process. Each edge in the graph has an arrow on it, pointing from a node whose output is used by the node pointed by the arrow as an input. The output $\delta^M_{t,T,K}$ at t is used as the input for next step $t+\Delta t$.

6.1.1 Difference Between GRU_δ And $\text{GRU}_{\text{TOTAL}}$

Comparing GRU_δ in Figure 4.1 and $\text{GRU}_{\text{TOTAL}}$ in Figure 6.1, it is clear that the model structures between GRU_δ and $\text{GRU}_{\text{TOTAL}}$ are similar. The most significant difference between $\text{GRU}_{\text{TOTAL}}$ and GRU_δ is that GRU_δ is built on minimizing discrete local risk as in section 2.1.1 while $\text{GRU}_{\text{TOTAL}}$ is built based on minimizing the discrete total risk as in in section 2.1.2.

More differences between $\text{GRU}_{\text{TOTAL}}$ and GRU_δ are given below:

- In GRU_δ, we have a local features vector $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$ which records local information at the hedging time t for hedging the option with expiry T and strike K. In Chapter 4 we demonstrate importance of using sequential learning by comparing performance with and without sequential features.

- Recognize importance of sequential features, in $\text{GRU}_{\text{TOTAL}}$, here we omit the local features vector $\mathbf{x}_t^{T,K} \in \mathbb{R}^{d_l}$ as the input to the model $\text{GRU}_{\text{TOTAL}}$ and use the sequential feature $\mathbf{Y}_t^{T,K}$ which contains all the sequential information between two consecutive rebalancing time t and $t-\Delta t$. Note that sequential feature also contains the local information the the hedging time t.

- Since the analytical formula for the variance-optimal total risk hedging [164] and spline total risk minimization formulation [50] both demonstrate the dependence of the current hedging position on the past hedging position, we include previous hedging position $\delta^M_{t-\Delta t,T,K}$ as the input to compute the current hedging position $\delta^M_{t,T,K}$.

The more details of the model structure of $\text{GRU}_{\text{TOTAL}}$ is discussed in Appendix D.

6.1.2 Training Objective For $\text{GRU}_{\text{TOTAL}}$

Assume that we are given a set of hedging scenarios identified by the expiry date T_i and strike K_i:

$$\{Scenario(T_1,K_1),\ldots,Scenario(T_M,K_M)\}$$

A natural total risk hedging loss function is the mean squared error:

$$MSE_{total} = \frac{1}{M}\sum_{i=1}^{M}(\text{Risk}^{total}_{t_0^i,T_i,K_i})^2$$

where $\text{Risk}^{total}_{t_0,T,K}$ is defined as in equation (6.1.1), $t_0^i = T_i - \frac{100}{250}$, and M is the number of hedging scenarios. A more appropriate criteria however is the relative hedging error instead of the absolute total hedging error since we are mixing scenarios of different strikes together (i.e., the scenarios include near-money, in-the-money, and out-of-the money options. The absolute hedging error for in-the-money options tend to be much bigger than those of near-the-money options

and out-of-money options). Additionally, MSE is sensitive to the existence of outliers. Furthermore, Coleman et al. [50] demonstrated that the l_1-norm error loss function can produce better hedging performance. Therefore, in training the $\text{GRU}_{\text{TOTAL}}$, we use the following objective:

$$Obj_{total} = \sum_{i=1}^{M} \left| \text{Rel}^{total}_{t_0,T_i,K_i} \right| \tag{6.1.2}$$

where the relative total hedging error is defined as:

$$\text{Rel}^{total}_{t_0,T,K} = \frac{D(t_0,T_i)\text{Risk}^{total}_{t_0,T,K}}{V_{t_0,T,K}} \tag{6.1.3}$$

6.1.3 Market Data Augmentation

As discussed in section 2.1.2, the final hedging portfolio value at T is:

$$\text{Risk}^{total}_{t_0,T,K} = \sum_{j=0}^{N_{rb}-1} \left\{ \left[\frac{S_{t_{j+1}}}{D(t_{j+1},T)} - \frac{S_{t_j}}{D(t_j,T)} \right] \delta^M_{t_j,T,K} \right\} + \frac{V_{t_0,T,K}}{D(t_0,T)} - V_{T,T,K} \tag{6.1.4}$$

where $D(t,T) = e^{-r(T-t)}$ is the discount factor and $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$ is the set of rebalacing time. The t_0 in this thesis is set to be N_H days to the expiry where N_H is a constant.

For total risk hedging, if we assume the starting hedging date t_0 is N_H days to expiry T, each data instance is then uniquely determined by the duplet $\{T,K\}$. If we only rely on market prices to build the total risk model, we will face the following challenges:

1. Options listed in an exchange only have fixed expiry dates and only a few strikes are listed every day. In other words, the number of unique duplet $\{T,K\}$ in actual market is small. For example, the expiration date for the standard S&P 500 index option is fixed at the third Friday of each month.[2] Therefore, if we only use market available expiration dates, the number of training scenarios will be severely limited.

2. In addition, the option with specific strike K and expiry T may not be traded on every trading date during its lifetime. Requiring the market option prices of strike K and expiry T to be available on entire hedging horizon $[t_0,T]$ is unrealistic especially for in-the-money and out-of-money options. Therefore, we will have to rely on certain parametric models calibrated to market prices to compute the necessary derived features (e.g., option sensitivity), which is used as the input of the data-driven model, when there are not market prices for the specific combination of T and K on some trading dates.

[2] Starting from 2016, CBOE also introduces weekly S&P 500 index options which expired on Monday, Wednesday and Friday of each week. The data we gathered is up to 2015-08-31, so our experiments in this thesis does not include those data.

3. Lastly, under an ideal setting, one should use non-overlapping underlying asset paths to generate training and testing hedging scenarios as one often observe autocorrelation in financial time series. In reality, using non-overlapping underlying asset path to generate training and testing scenarios is not practical. For instance, assuming we are building a data-driven model for hedging 3 months until expiry, with non-overlapping underlying asset paths, 20 years of market data can provide only 80 different non-overlapping underlying asset paths, making building a data-driven model difficult.

In order to overcome above challenges, we propose the following remedy.

1. Overlapping underlying asset paths are allowed.
2. A no-arbitrage surface is calibrated on each business day to match the market prices. We will query the calibrated surface to obtain the option prices and option sensitivities when the corresponding market data is not available.
3. Instead of using only market available expiration dates, we assume every business day can be the expiration date of the options. The option prices and option sensitivities for these newly added expiry dates will come from querying the no-arbitrage surface.

Therefore, we greatly increase the number of training scenarios, since now the hedging scenarios can have more combinations of T and K even if the combinations of T and K are not directly observed in the market. We create a parametrization of the price surface that is **arbitrage-free**. This is done by using SABR model to match the market volatility smiles and we use an arbitrage-free interpolation based on Local Volatility Function (LVF) model to interpolate the SABR model value between expiries.

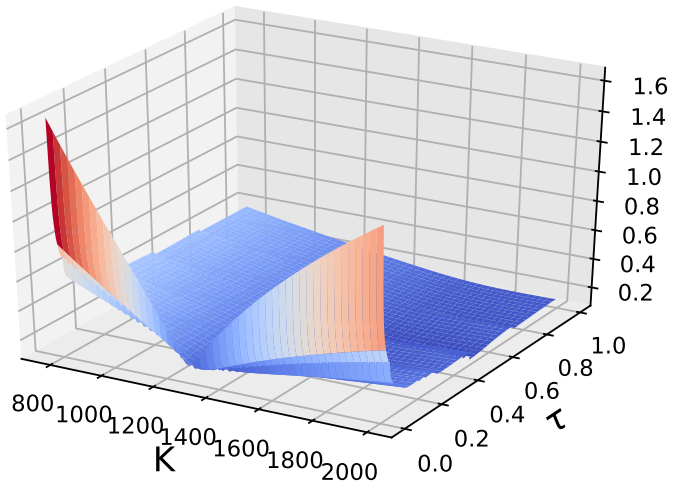
As an example, in Figure 6.2, we show the resulting price surface and implied volatility surface for SP500 index call options on 2012-01-04. In Figure 6.2, we use $\tau = T - t$ to denote the time to maturity.

We refer an interested reader to Appendix §F for details of no arbitrage surface calibration.

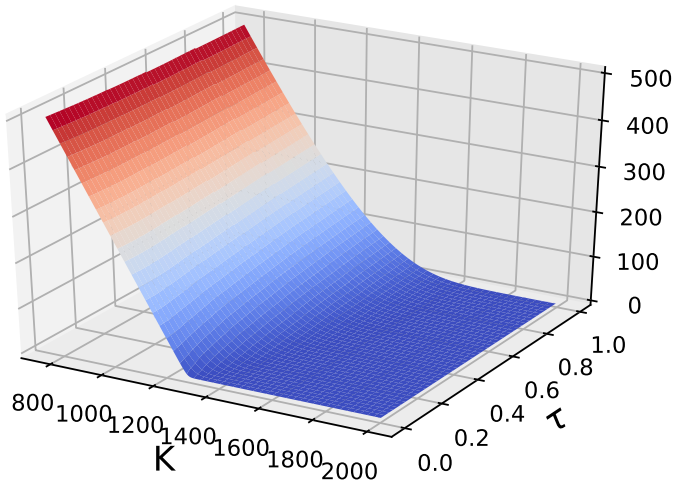
6.1.4 Training and Testing Data Construction

In this section, we will discuss the training and testing data construction for building the total hedging model with the surfaces constructed from SABR and LVF model. The training and testing data set are the collection of hedging scenarios. Each of the hedging scenarios is identified uniquely by a duplet of expiry date and strike (T,K). [3] Let t_0 be the initial date to set up the hedging portfolio and $\mathbf{t}_B = \{t_0,\dots,T\}$ be the set of all business days in between the initial date t_0 and expiry date T. Each of the hedging scenarios is the collection of the following time series identified uniquely by (T,K):

[3] If we fixed the gap between two rebalancing time to be Δt and we fix the number of times we rebalance the hedging portfolio to be N_{rb}, then given an expiry T, one can easily deduct the rebalance time $\{t_0,t_1,\dots,t_{N_{rb}-1}\}$. Therefore, we can uniquely define a hedging scenario just by T and K.



(a) Implied Volatility Surface



(b) Price Surface

Figure 6.2: The price surface and implied volatility surface calibrated to SP500 index call options on 2012-01-04. The smile is more pronounced for the shorter maturities than the longer maturities, which is consistent with observations from various studies [34, 157] using market data. Interested readers can also refer to [157] for some mathematicaly explaination on why the smile becomes more and more flattened as τ increases.

- $\{S_t|\forall t \in \mathbf{t}_B\}$: the the time-series of underlying prices with t_0 as the initial date and T as the expiry date.
- $\{V_{t,T,K}|\forall t \in \mathbf{t}_B\}$: the time-series of option value for a hedging scenario identified by (T,K).
- $\{\mathbf{y}_t^{T,K}|\forall t \in \mathbf{t}_B\}$: the time-series of feature vectors for a hedging scenario identified by (T,K).

6.1.4.1 Construction of Training Scenarios

Based on the calibration process we discussed above, we summarize the procedure for generating training scenarios for hedging N_H business days as the following.

1. Each of the hedging scenarios is identified uniquely by a duplet of expiry and strike (T,K). We need to construct the time series corresponding to the duplet as the input to our model. For a training scenario, we do not require the option prices for the duplet (T,K) to exist in market. We will query the surface constructed by Algorithm 5 for each time t: $V^t_{model}(T,K)$ for option values and compute the associated option related sensitivities, which are used to construct the feature vector for the model. In this way, we greatly increase the number of training scenarios.
2. The starting date t_0 to set up the initial hedging portfolio is N_H-business days away from the expiry date T. In this thesis, N_H is set to be 100. We assume there are 250 business days in a year. Denote $T^{mkt}_{t,max}$ as the maximum expiry listed in exchange at time t. We further comment that we always have $T^{mkt}_{t,max} - t > 100/250$ in market for all the business dates we include in the experiments. Therefore, no volatility extrapolation is needed.
3. We check all the market observed strikes for all the business dates between t_0 and T. Let $K^{mkt}_{max}(t_0,T)$ be the maximum of strikes we observed in market between t_0 and T. The grid of strikes for the outputting option values with expiry date T is defined as: $\mathbf{K}_{grid}(t_0,T) = \{0 = K_0 < K_1 < \cdots < 2 * K^{mkt}_{max}(t_0,\widehat{T})\}$ where $K_i - K_{i-1} = \Delta K, i \geq 1$. In this chapter, we set $\Delta K = 5$ for experiments on S&P 500 index options which is consistent with the S&P500 index option strike specification in real market [113].

A detailed description is given in Algorithm 6 in Appendix §**??**.

6.1.4.2 Construction of the Testing Scenarios

The key differences between the testing scenarios and the training scenarios are:

- For testing scenarios, we use real market prices to initialize the hedging portfolio at t_0 and the strikes and expiries are real market strikes and expiries.

- For testing scenarios, we use real option market data to construct the time-series whenever associated option market data is available. Otherwise, we will query the calibrated option value function.
- For training scenarios, we use model prices to initialize the hedging portfolio. The strikes and expiries do not necessarily have to be are real market strikes and expiries.
- For training scenarios, we always query the parametrization of the option value we calibrated to construct the time series. Please note that since we calibrate the models to match the market prices, when market prices are available, the model prices will be very close to the market prices.

A summary of the construction of testing scenarios is as the following. A detailed algorithm is given in Algorithm 7.

1. A testing expiry date T must be a real expiry date that exists in market. Before 2016, For the S&P 500 index options listed in Chicago Board Options Exchange (CBOE) the expiration dates are the third Fridays of each month. After 2016, more expiration dates are introduced in CBOE.
2. The date t_0 to set up the initial hedging portfolio is N_H-business days away from T.
3. On the starting date t_0, we can obtain all market option prices for the expiry T and we have a grid of market strikes for expiry T on date t_0: $\mathbf{K}^{mkt}_{grid}(t_0,T) = \{K^{mkt}_{t_0,T,1},\ldots,K^{mkt}_{t_0,T,N_K}\}$. Note that we have market options prices for all $K \in \mathbf{K}^{mkt}_{grid}(t_0,T)$ at time t_0.
4. On each business day t in between t_0 and T, an arbitrage free surface is constructed $\{V^t_{model}(T,K)\}_{T,K}$ using Algorithm 5. When there is no market price $V^{mkt}_{t,T,K}$ on time t for $K \in \mathbf{K}^{mkt}_{grid}(t_0,T)$, we will query $\{V^t_{model}(T,K)\}_{T,K}$ to obtain option prices and option sensitivities. Note that, with this approach, for instance, a part of the time series of option prices for a testing scenario can be real market prices while the other part can be model prices from the parametrization obtained following the calibration process in section F.

6.1.4.3 Construction of Training, Testing and Validation Data Sets

For the experiments in this chapter, we are hedging for a relatively long period (100 business days) until the expiry. Empirically, we have observed that if we do not update the data-driven model during the hedging period, the performance from the data-driven hedging would be much worse than the performance of the traditional parametric hedging models such as hedging with delta produced by Black-Scholes implied volatility. This is not surprising since market can drastically change during the hedging period which is relatively long and we cannot assume one set of parameter for a data-driven model to be effective for such long period. Lastly, by comparing the performance of local risk model $\mathrm{DKL}_{\mathrm{SPL}}$, which is updated on a monthly basis, with those of

NN_δ and GRU_δ ,which are updated on a daily basis in Chapter 5, we can already see the effectiveness of more frequent update. Therefore, the training and validation data set are updated as we move from one rebalancing date to another rebalancing date. We denote the total risk hedging model as $\text{GRU}_{\text{TOTAL}}$. The detailed description of the model $\text{GRU}_{\text{TOTAL}}$ will be discussed in the following section 6.1.

A summary of the construction procedure is given as the following. The detailed procedure of the construction of training, testing and validation dataset and the overall model building procedure is given in Algorithm 8 in Appendix **??**.

6.1.5 Training Procedure For $\text{GRU}_{\text{TOTAL}}$

We initialize the GRU parameters using the same procedure as in section 4.2.1 and we pretrain the $\text{GRU}_{\text{TOTAL}}$ similarly as in section 4.2.3. Early stopping is used as the regularization techniques. We reserve the validation set to determine when to stop the training. We train $\text{GRU}_{\text{TOTAL}}$until trust region algorithms (i.e., Algorithm 1) stops and slect the best performing model parameters on validation set based on the the total risk objective (6.1.2). The parameters for trust region algorithm is in Table 4.1. Early stopping is used as the regularization, which is the same as in chapter 4. The overall model building procedure is given in Algorithm 8.

6.2 Total Discrete Hedging Performance Comparison Using S&P 500 index Options

Using the S&P 500 (European) index option market data from September 1, 1996 to August 31, 2015[4], we compare the total hedging performance of different hedging strategies. We evaluate the total hedging performance using the following 5 criteria:

1. The mean absolute value of the relative hedging error:

$$Mean_{(t_0,T,K)}\left(\left|\text{Rel}^{total}_{t_0,T,K}\right|\right)$$

   for all the testing scenarios.

2. The 95% Value-at-Risk (VaR) of the relative total hedging error $\text{Rel}^{total}_{t_0,T,K}$

3. The 95% Conditional-Value-at-Risk (CVaR) of the relative total hedging error $\text{Rel}^{total}_{t_0,T,K}$

4. The 99% Value-at-Risk (VaR) of the relative total hedging error $\text{Rel}^{total}_{t_0,T,K}$

5. The 99% Conditional-Value-at-Risk (CVaR) of the relative total hedging error $\text{Rel}^{total}_{t_0,T,K}$

[4]The option historical data from OptionMetric [147] started on September 1, 1996. Due to the limits of data license, we only have access to OptionMetric up to August 31, 2015.

6.2.1 Data and Experimental Setting

The sequential inputs to $\text{GRU}_{\text{TOTAL}}$, $\mathbf{Y}_t^{T,K}$, at a rebalancing time t, are the time series recorded daily from previous rebalancing time $t - \Delta t$ to current rebalancing time t for the following features:

Option price
Black–Scholes implied volatility
Black–Scholes delta
Black–Scholes vega
Bartlett delta
Time to expiry
S&P 500 index price
VIX index price
Moneyness S/K
Minimum variance delta δ_{MV} (2.3.5)
Strike K

Table 6.1: Sequential features for $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$ at time t are the time series of features listed in this table. The time series are constructed according to the procedures as in Algorithm 6 and Algorithm 7.

The number of hidden states, for the single-layer GRU encoder, the neural network outputting $\widehat{\delta}_{t,T,K}^{M}$, and the neural network outputting W_δ in Figure 6.1, are all set to be 5. Specifically, we compare with the following methods,

- $\text{GRU}_{\text{TOTAL}}$: the model shown in Figure 6.1 and is trained with the total risk objective (6.1.2).
- Bartlett: Barlett corrective delta based on (2.3.18),
- BS: Black–Scholes delta based on the implied volatility

The hedging period is fixed to be 100 business days. We have two different hedging frequencies: weekly and monthly hedging. For weekly hedging, we rebalance every 5 business days so the number of rebalacing times is $N_{rb} = 20$. For monthly hedging, we rebalance every 20 business days so the number of rebalacing times is $N_{rb} = 5$.

6.2.2 Call Option Total Hedging Comparison

In this subsection, we present the results for call options. We show the hedging performance for Near-The-Money(NTM), In-The-Money(ITM), Out-of-The-Money(OTM) separately. Note that we are not training models for NTM, ITM and OTM separately. We still train the model using all

training set. The NTM, OTM, and ITM scenarios are classified based on the Black-Scholes delta at the initial date t_0 where we set up the hedging portfolio: $\delta^{BS}_{t_0,T,K}$. For call option, the criteria is:

- NTM: $0.3 \leq \delta^{BS}_{t_0,T,K} < 0.7$
- ITM: $0.7 \leq \delta^{BS}_{t_0,T,K} < 0.95$
- OTM: $0.05 \leq \delta^{BS}_{t_0,T,K} < 0.3$

We omit the testing scenarios for deep in-the-money and deep out-of-the money options due to the fact that they are highly illiquid in market and their market quotes are highly unreliable. Also, the deep in-the-money and deep out-of-the money scenarios are deleted from training set and validation set.

- Deep ITM: $0.95 \leq \delta^{BS}_{t_0,T,K} < 1.0$
- Deep OTM: $0.0 \leq \delta^{BS}_{t_0,T,K} < 0.05$

6.2.2.1 Call Option Weekly Hedging Comparison

In Table 6.2, we demonstrate the results on weekly hedging call options. Furthermore, in Figure 6.3, and Figure 6.4, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the BS model and the Bartlett model respectively.

From Table 6.2, we can see that, $\text{GRU}_{\text{TOTAL}}$ performs better than the other methods in terms of mean absolute relative hedging error for NTM and OTM scenarios, except that BS model performs slightly better than $\text{GRU}_{\text{TOTAL}}$ for ITM scenarios. The difference between $\text{GRU}_{\text{TOTAL}}$ and BS model is small for ITM scenarios. Additionally, $\text{GRU}_{\text{TOTAL}}$, in most of cases, perform the best in terms of VaR and CVaR, indicating that $\text{GRU}_{\text{TOTAL}}$ performs better in reducing the tail loss. The exceptions is given as the following:

- BS model performs best in terms of VaR(99%) and CVaR(99%) for OTM scenarios. This is an interesting observation. From 6.3 (c), We also note that the extreme tail on the profit side from BS model on OTM scenarios is actually longer than the other three models, indicating a larger probability of getting profit. However, we notice that the difference in CVaR(99%) for OTM scenarios between $\text{GRU}_{\text{TOTAL}}$ and BS model is small.
- Bartlett model performs the best in terms of CVaR(99%) for ITM scenarios. However CVaR(99%) of $\text{GRU}_{\text{TOTAL}}$ is only slightly worse than that of the Bartlett model meanwhile $\text{GRU}_{\text{TOTAL}}$ perform still the best in terms of the VaR(99%).

Another interesting observation is that Bartlett delta actually performs worse than BS delta in most of the cases as shown in Table 6.2. We suspect that this is due to the fact that SABR model

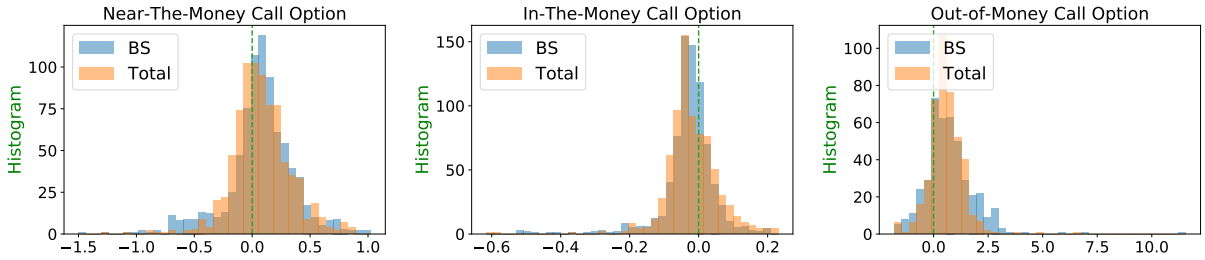


Figure 6.3: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and BS Model on weekly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

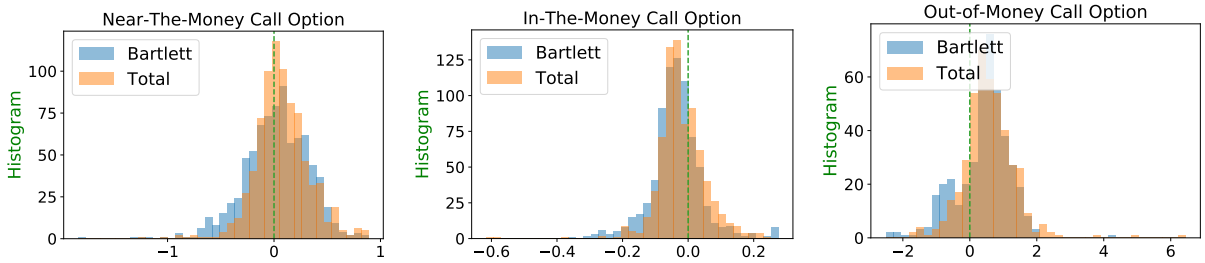


Figure 6.4: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and Bartlett model on weekly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	GRU_{TOTAL}	**0.1927**	0.0571	**0.7344**
	Bartlett	0.2347	0.0641	0.7383
	BS	0.2198	**0.0531**	0.9706
VaR (95%)	GRU_{TOTAL}	**0.2827**	**0.1121**	**0.5298**
	Bartlett	0.4836	0.1656	0.9841
	BS	0.4823	0.1523	0.8603
CVaR (95%)	GRU_{TOTAL}	**0.4721**	**0.1865**	**1.0003**
	Bartlett	0.7103	0.2192	1.4232
	BS	0.7009	0.2724	1.2299
VaR (99%)	GRU_{TOTAL}	**0.5301**	**0.1976**	1.5077
	Bartlett	0.778	0.2654	1.6152
	BS	0.7171	0.3653	**1.3363**
CVaR (99%)	GRU_{TOTAL}	**0.8205**	0.3261	1.6090
	Bartlett	1.0827	**0.2883**	2.1225
	BS	1.0040	0.4712	**1.6074**

Table 6.2: Summary of weekly hedging S&P 500 call options (testing set) for 100 business days with total hedging evaluation criteria described in section 6.2. Please note that the total hedging evaluation in this table assumes we are at the sell-side of the option trading.

was originally designed for modeling interest rate derivatives, the time to maturity for which is usually bigger than one-year, and it is less suitable to model option surface with extreme short time to maturity[35]. For weekly hedging, we have used SABR model to produce Bartlett delta with extremely small time to maturity, e.g., 5/250 for the last rebalancing time. Notice that in chapter 5 when we compare models on local risk criteria, options with time-to-expiry less than 14 days are removed from the data set. Therefore, we did not notice this phenomenon.

6.2.2.2 Call Option Monthly Hedging Comparison

In Table 6.3, we demonstrate the results on monthly hedging call options. Furthermore, in Figure 6.5 and Figure 6.6, we compare the distribution of the relative hedging error with the distributions of the BS model and Bartlett model respectively.

From Table 6.3, we can see that, GRU_{TOTAL} performs better than the other methods in terms of mean absolute relative hedging error for NTM and ITM scenarios. Bartlett method performs best in terms of mean absolute relative hedging error for OTM scenarios. In terms of VaR and CVaR, by comparing Table 6.3 and Table 6.2, GRU_{TOTAL} is less dominant in monthly hedging call options than in weekly hedging call options. Bartlett delta produces best VaR(99%) and CVaR(99%) for NTM scenarios. However, from Table 6.3 we can also see, the performance of GRU_{TOTAL} is very close to best performance even if GRU_{TOTAL} is not the dominant model in terms of certain criteria.

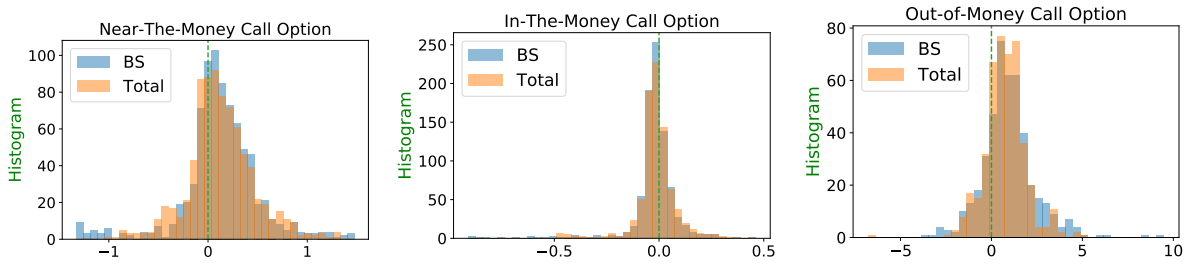


Figure 6.5: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and BS model on monthly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

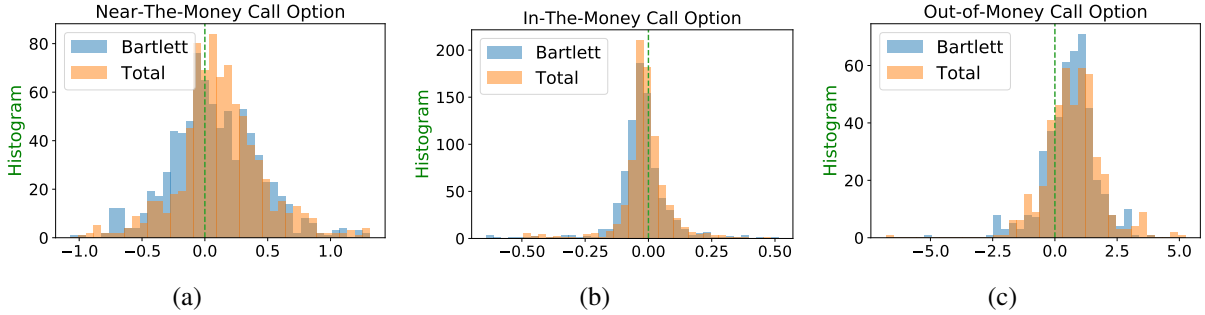


Figure 6.6: Comparing total risk model $\text{GRU}_{\text{TOTAL}}$ and bartlett model on monthly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are at the sell-side of the option trading.

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	GRU_{TOTAL}	**0.2643**	**0.0633**	1.0479
	Bartlett	0.282	0.073	**0.9674**
	BS	0.2865	0.0655	1.3248
VaR (95%)	GRU_{TOTAL}	**0.4102**	**0.1472**	**1.0842**
	Bartlett	0.4775	0.1611	1.1936
	BS	0.5115	0.1554	1.3442
CVaR (95%)	GRU_{TOTAL}	**0.6073**	**0.3125**	**1.6658**
	Bartlett	0.6680	0.3372	2.0221
	BS	0.9735	0.4380	2.0016
VaR (99%)	GRU_{TOTAL}	0.7752	**0.4300**	**1.7567**
	Bartlett	**0.7201**	0.4815	2.3799
	BS	1.2384	0.6058	2.8419
CVaR (99%)	GRU_{TOTAL}	0.8692	**0.4627**	**2.7536**
	Bartlett	**0.8090**	0.5725	2.8797
	BS	1.2864	0.7859	3.0839

Table 6.3: Summary of monthly hedging S&P 500 call options (testing set) for 100 business days with total risk hedging evaluation criteria described in section 6.2. The total hedging evaluation in this table assumes we are on the sell-side of the option trading.

6.2.3 Put Option Total Risk Hedging Comparison

In this subsection, we present the results for put options. We again show the hedging performance for Near-The-Money(NTM), In-The-Money(ITM), Out-of-The-Money(OTM) separately. The NTM, OTM, and ITM scanerios are classified based on the Black-Scholes delta at the initial date t_0 where we set up the hedging portfolio: $\delta^{BS}_{t_0,T,K}$. For put option, the criteria is:

- NTM: $-0.3 \geq \delta^{BS}_{t_0,T,K} > -0.7$
- ITM: $-0.7 \geq \delta^{BS}_{t_0,T,K} > -0.95$
- OTM: $-0.05 \geq \delta^{BS}_{t_0,T,K} > -0.3$

We omit the testing scenarios for deep in-the-money and deep out-of-the money options due to the fact that they are highly illiquid in market and their market quotes are highly unreliable. Also, the deep in-the-money and deep out-of-the money scenarios are deleted from training set and validation set.

- Deep OTM: $0.0 \geq \delta^{BS}_{t_0,T,K} > -0.05$
- Deep ITM: $-0.95 \geq \delta^{BS}_{t_0,T,K} > -1.0$

6.2.3.1 Put Option Weekly Hedging Comparison

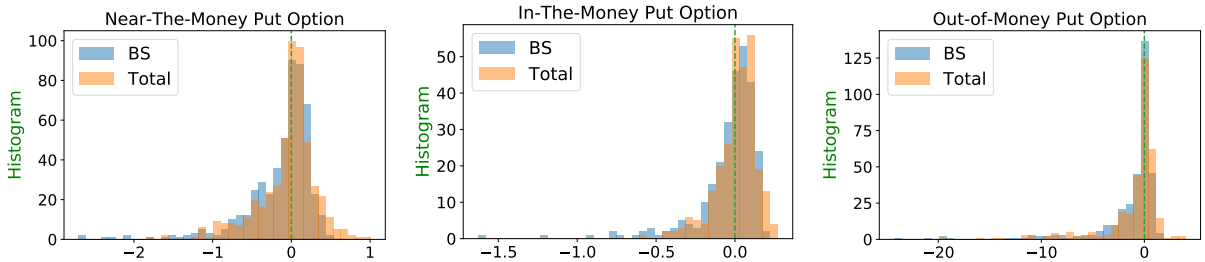
In Table 6.4, we demonstrate the results on monthly hedging put options. Furthermore, in Figure 6.8 and Figure 6.7, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the relative hedging error of the BS model and the Bartlett model respectively.

From Table 6.4, we can see that, $\text{GRU}_{\text{TOTAL}}$ performs better for weekly hedging put options in terms of most of the criteria for NTM, ITM, and OTM options. There is one exceptions: Bartlett delta performs slightly better than $\text{GRU}_{\text{TOTAL}}$ for OTM scenarios in terms of mean absolute relative hedging error.

Another interesting observation is that, for put options, the loss tail of the relative hedging distribution is significantly longer than call option. We suspect this is due to the fact that selling put options during market crisis period can lead to significant losses as the original OTM options can become ITM in a short period of time, especially when we are getting closer to the expiry. For weekly hedging put options, it is worth to note that the tail loss, which is measured by VaR and CVaR, from $\text{GRU}_{\text{TOTAL}}$ is significantly smaller than those from the BS model and the Bartlett model.

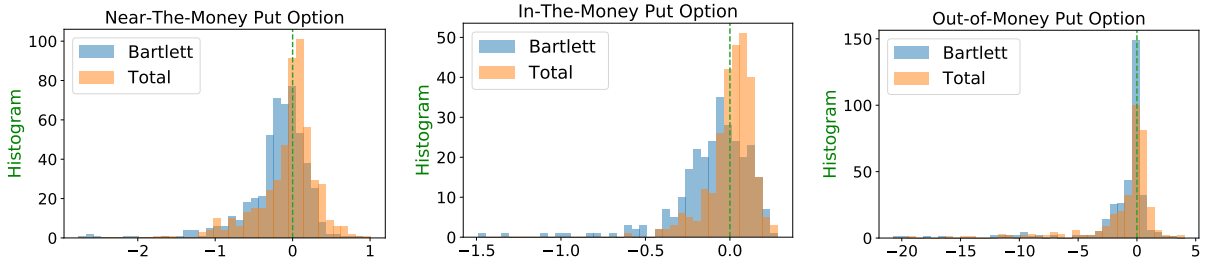
		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	**0.2535**	**0.0965**	1.5356
	Bartlett	0.2993	0.167	**1.4815**
	BS	0.2773	0.1227	1.7109
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.8124**	**0.2364**	**7.2478**
	Bartlett	0.9374	0.5133	8.5614
	BS	0.8854	0.4274	8.7374
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**1.0475**	**0.3452**	**10.9438**
	Bartlett	1.4781	0.8078	12.2226
	BS	1.4812	0.7236	13.3299
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**1.1138**	**0.3763**	**11.7573**
	Bartlett	1.6118	0.99	12.2933
	BS	1.7625	0.7979	19.0822
CVaR(99%)	$\text{GRU}_{\text{TOTAL}}$	**1.3597**	**0.4616**	**15.1555**
	Bartlett	2.3355	1.2264	17.4385
	BS	2.2831	1.1347	20.6413

Table 6.4: Summary of weekly hedging S&P 500 put options (testing set) for 100 Business days with total risk hedging evaluation criteria described in section 6.2. Please note that the total hedging evaluation in this table assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure 6.7: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and BS model on weekly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure 6.8: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and Bartlett model on weekly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

6.2.3.2 Put Option Monthly Hedging Comparison

In Table 6.5, we demonstrate the results on monthly hedging put options. Furthermore, in Figure 6.9 and Figure 6.10, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the BS model and the Bartlett model respectively.

From Table 6.5, we can see that, $\text{GRU}_{\text{TOTAL}}$ is still the dominant method for monthly hedging put options in terms of most of the criteria for NTM, ITM, and OTM scenarios. However, by comparing Table 6.4 and Table 6.5, we can see $\text{GRU}_{\text{TOTAL}}$ is less dominant in monthly hedging than in weekly hedging. In certain case, Bartlett methods can perform much better. For instance, for NTM scenarios, the CVaR (95%) and CVaR (99%) from Bartlett method is significantly better than the other two comparing methods. Another interesting observation is that the $\text{GRU}_{\text{TOTAL}}$

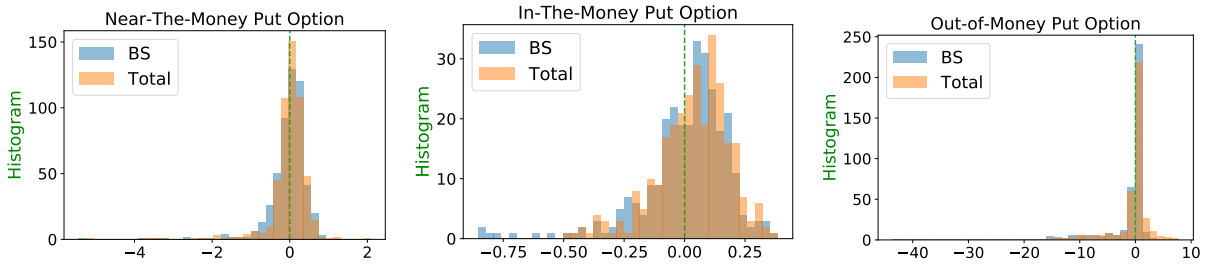
		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	**0.2986**	**0.1240**	1.7639
	Bartlett	0.3205	0.1583	**1.7383**
	BS	0.3224	0.1342	1.8482
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.7395**	**0.2562**	**8.5602**
	Bartlett	0.8370	0.3583	9.0303
	BS	0.7768	0.3088	9.7018
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	1.7761	**0.3577**	**13.3160**
	Bartlett	**1.5710**	0.6016	14.4425
	BS	1.8682	0.5401	16.1024
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**2.1792**	**0.4121**	15.2323
	Bartlett	2.1925	0.7728	**15.0144**
	BS	2.5569	0.7583	15.8393
CVaR (99%)	$\text{GRU}_{\text{TOTAL}}$	3.4001	**0.4509**	**20.6503**
	Bartlett	**2.9164**	0.8463	24.1757
	BS	3.5486	0.8109	26.4941

Table 6.5: Summary of monthly hedging S&P 500 put options for 100 business days with total hedging evaluation criteria described in section 6.2. The total hedging evaluation in this table assumes we are on the sell-side of the option trading.

produces longer tail for NTM and OTM scenarios on the profit side while for ITM scenarios, $\text{GRU}_{\text{TOTAL}}$ has a much shorter tail on the loss side.

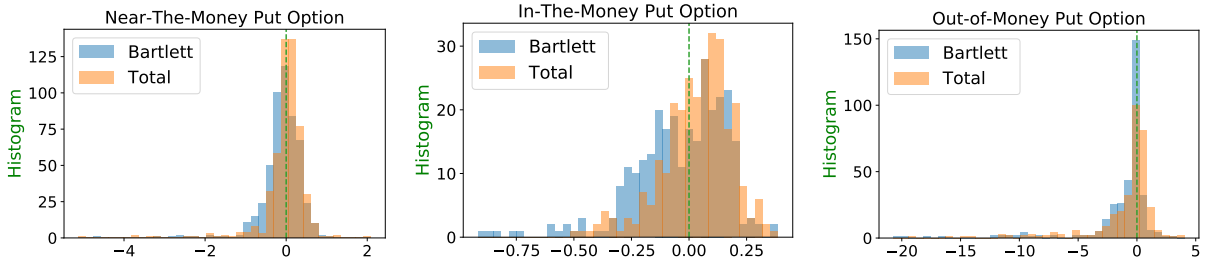
6.2.4 Comparison to Local Risk Hedging Model

Since GRU_{δ} is built on top of pure market data while $\text{GRU}_{\text{TOTAL}}$ is built on top of augmented market data, we will not compare the performance of GRU_{δ} and $\text{GRU}_{\text{TOTAL}}$ directly. Since, here we are more interested in the effect of the choice of objective functions. Therefore, we define a new comparing model $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$. The model structure of $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$ is exactly the same as



(a) (b) (c)

Figure 6.9: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and BS model on monthly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure 6.10: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and Bartlett model on monthly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

$\text{GRU}_{\text{TOTAL}}$ which is as in Figure 6.1. The only difference is $\text{GRU}_{\text{TOTAL}}$ is built on minimizing objective (6.1.2), while the $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is built on minimizing a local risk objective.

More specifically, for an expiry T and a strike K, at a rebalancing time t_j ,we have

$$\begin{aligned}\Delta V_{t_j,K,T} &= D(t_0,t_{j+1})V_{t_{j+1},K,T} - D(t_0,t_j)V_{t_j,K,T}\\ \Delta S_{t_j} &= D(t_0,t_{j+1})S_{t_{j+1}} - D(t_0,t_j)S_{t_j}\\ D(t,T) &= e^{-r(T-t)}\end{aligned}$$

$$t_j = t_0 + j\Delta t;\ j = 0,\ldots,N_{rb}-1;\ \ t_0 = T - N_{rb}\Delta t\ .$$

The objective for the $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is therefore:

$$Obj_{Local} = \sum_{i=1}^{M}\sum_{t\in\mathbf{t}^i_{RB}} |\Delta V^{mkt}_{t,T^i,K^i} - \Delta S_t \delta^M_{t,T^i,K^i}| \tag{6.2.1}$$

where $\mathbf{t}^i_{RB} = \{t^i_0,\ldots,t^i_{N_{rb}-1}\}$ is the set of rebalancing dates for the i-th hedging scenarios with expiry T^i and initial date t^i_0. The model structure for $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is the same which is discussed in Appendix D. The same set of hedging scenarios are used as the training, testing and validation data sets. The training procedure is also the same as indicated in Algorithm 8. The only difference is the objective function used in training.

The detailed comparison between $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is given in Appendix E. Here we summarize the major results:

- For weekly hedging call options, $\text{GRU}_{\text{TOTAL}}$ performs better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of most of total risk measures. However, in terms of tail loss reduction, the improvement from $\text{GRU}_{\text{TOTAL}}$ over $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is less significant.
- For monthly hedging call options, $\text{GRU}_{\text{TOTAL}}$ improve over $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of the mean absolute relative error slightly. The $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ perform roughly the same in terms of tail loss measured by VaR and CVaR.
- For weekly hedging put options, $\text{GRU}_{\text{TOTAL}}$ still performs better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of reducing the mean absolute relative error for ITM and OTM scenarios and the performance for NTM scenarios is similar. On the other hand, in terms of tail loss measured by VaR and CVaR, the reduction from $\text{GRU}_{\text{TOTAL}}$ over $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is significant.
- For monthly hedging put options with NTM and ITM scenarios, we achieve better mean absolute relative error from $\text{GRU}_{\text{TOTAL}}$. For OTM scenarios, $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ performs better in terms of mean absolute relative error. The tail loss measured by VaR and CVaR for NTM scenarios is roughly the same for $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. The tail loss from $\text{GRU}_{\text{TOTAL}}$ for OTM scenarios is slightly better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. The tail loss from $\text{GRU}_{\text{TOTAL}}$ for ITM scenarios is significantly better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$.

As we have already discussed in section 2.1.2, with the assumption of zero interest rate, the *discrete total hedging risk* which is defined as:

$$\text{Risk}^{total}_{t_0,T,K} = \sum_{j=0}^{N_{rb}-1} \left\{ \Delta S_{t_j} \delta_{t_j,T,K} - \Delta V^{mkt}_{t_j,T,K} \right\} = \sum_{j=0}^{N_{rb}-1} \text{Risk}^{local}_{t_j,T,K} \tag{6.2.2}$$

In other words, *discrete total hedging risk* is the summation of the *discrete local hedging risk* evaluated at discrete rebalancing time $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$. As a consequence, building a model reducing the discrete local hedging risk will reduce the discrete total hedging risk as well. Therefore, it is not surprising to see that $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is still competitive in terms of total risk measurements.

Chapter 7

Conclusion and Future Work

In this thesis, we have proposed a direct kernel hedging model DKL_{SPL} and a novel encoder-decoder model, GRU_{δ} , for discrete local risk option hedging. The DKL_{SPL} is our first exploration on computing a data-driven local hedging model without estimating a option pricing model. GRU_{δ} is proposed to further improve direct data-driven local risk hedging using machine learning techniques. GRU_{δ} consists of an encoder, which generates a concise representation of the past market information. The decoder uses the Black-Scholes delta as a pre-trained model and utilizes a gate to generate a predicative hedging model, combining the pre-trained delta model and the outputs from the encoder. Feature selections are implemented through normalized weights embedded in the model training. In addition, a data instance adaptive Huber loss function is incorporated for robustness, with the error from the pre-trained Black-Scholes delta model for that instance as the thresholding parameter.

Using the S&P 500 index and the index option data, from January 1, 2004, to August 31st, 2015, we assess and compare hedging performance of the GRU_{δ} and DKL_{SPL} with other hedging strategies in terms of local risk criteria. For weekly and monthly hedging, computational results demonstrate that performance of the proposed GRU_{δ} significantly surpasses that of the MV model, SABR-Bartlett, regularized spline kernel model DKL_{SPL}, all of which perform significantly better than the Black–Scholes model with implied volatility in terms of local risk hedging criteria (6.2.1). In addition, the DKL_{SPL} also outperforms MV model in terms of weekly and monthly hedging results. We further demonstrate that the encoder for the sequential features plays a significant role in GRU_{δ}, since removing the encoder deteriorates hedging performance. Lastly, by comparing the weekly and monthly hedging performance from the GRU_{c}, for which we remove the output gate and training with MSE, and GRU_{δ}, we demonstrate that the output gate, robust loss function and also play significant roles in GRU_{δ}.

We further demonstrate that the daily hedging performance of the proposed GRU_{δ} also surpasses that of the MV hedging method, LVF and SABR corrective methods (implemented in [112]), data-driven regularized spline kernel network model DKL_{SPL}, and SABR-Bartlett. In addition, DKL_{SPL} also outperforms MV hedging method , LVF and SABR corrective methods (implemented in [112]).

In addition, from the testing hedging performance, we assess feature importance in GRU_δ for the S&P 500 index option hedging. The monthly average feature weights identify the underlying as the most important local feature and the past implied volatility sequence as the most important sequential feature during normal market periods.

To assess whether minimizing the total risk directly can lead to better performing total risk minimization strategy, in the context of data-driven hedging, we extend GRU_δ to $\text{GRU}_{\text{TOTAL}}$ for multi-step total risk hedging. To deal with the challenges of acquiring enough market option information for building data-driven hedging models, we augment the market data using SABR model and local volatility model. Using the S&P 500 index option market data from January 1st, 2000 to August 31st, 2015, we compare the weekly and monthly hedging performance of the proposed total risk hedging model $\text{GRU}_{\text{TOTAL}}$ with the sequential data-driven local risk hedging model $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$, which adopts the same model structure but is trained with a local risk training objective, the Black-Scholes delta with implied volatilities, and the SABR-Bartlett delta. We measure the total risk hedging performance, which is evaluated on the expiries of the options. We demonstrate the effectiveness of the total risk hedging model $\text{GRU}_{\text{TOTAL}}$ in reducing the sell-side loss tail risk for both put and call options. We also confirm that the total hedging model $\text{GRU}_{\text{TOTAL}}$ often leads to better hedging performance in terms of total hedging criteria when compared with $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$, SABR-Bartlett method, and Black-Scholes model. However, alternative local risk model $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$ remain competitive in controlling the total hedging risk.

The main objective of this research is to assess hedging performance of strategies learned directly from the historical time series of the market option price and underlying price, using machine learning methods. Hedging performance comparisons between data-driven models DKL_{SPL}, GRU_δ, $\text{GRU}_{\text{TOTAL}}$,and $\text{GRU}_{\text{TOTAL}}^{\text{LOCAL}}$ and the classical option hedging based on parametric model calibration suggest that the data driven learning hedging can be a viable alternative to the classical methods, potentially leading to better hedging performance.

In terms of limitation of this research, we compare the data-driven models mostly with parametric models available in academic literature. We understand that the actual industry practice may not apply the parametric models in the same way as we did in this thesis for hedging derivatives. While it would also be interesting to compare hedging methods actually adopted in the financial industry, limitation in accessing industry practice makes it difficult to conduct such a study.

Additionally, comparing with the calibration of the traditional parametric pricing models on vanilla index options, the learning process for the data-driven hedging model is less computationally efficient. However, one should notice that once the model learning of the data-driven model is done, the outputing process of the hedging position is computaionally efficient. In practice, one can tra the model after the business trading hours and use the trained model to produce the hedging position efficiently during the business trading hours.

Lastly, the learning process requires certain amount of historical data. For calibrating a parametric model, one usually needs much less data and can build the model based on market data observed on spot and compute the sensitivity as hedging position accordingly. Therefore, our proposed data-driven models are not directly applicable to the illiquid derivative markets.

For extending our work, we note that there are several directions:

- In this thesis, we rely on market data to generate the hedging scenarios. The volatility surface is calibrated from market data and underlying price paths are extracted from market with overlapping period. Our models based on a neural network approach have less parameters compared with other applications of the deep learning techniques, given the relative scarcity of available historical data. For future work, one can explore how we can use machine learning techniques to generate hedging scenarios so that we can combine artificial scenarios with real scenarios. This will enable us to build more complex model for hedging. Indeed, for this direction, there are already several attempts. For example, Bergeron et al. [18] apply the variational autoencoders [121] on generating synthetic volatility surface that are indistinguishable from those observed historically. Pardo and López [149] apply the Generative Adversarial Networks (GANs) [91] on learning the underlying structure inherent to the dynamics of financial series and acquiring the capacity to generate scenarios that share many similarities to those seen in the historic time series.

- In this thesis, we use S&P 500 index options for experimental comparison. It will be interesting to explore effectiveness of the data-driven models on more complex derivatives such as basket options where the dimensionality of the underlying is higher.

- In this thesis, we have not included transaction cost into our models. A more realistic model should include the effect of the transaction cost as in [28].

- In this thesis, for total hedging model, we assume we rebalance every 5 business days or 20 business days. In reality, we do not have to fix the interval length between two rebalancing times. We can extend the model so that the data-driven model determines when is the best time to rebalance the hedging portfolio.

References

[1] Yacine Aıt-Sahalia and Jefferson Duarte. Nonparametric option pricing under shape restrictions. *Journal of Econometrics*, 116(1):9–47, 2003.

[2] Yacine Aït-Sahalia and Andrew W Lo. Nonparametric estimation of state-price densities implicit in financial asset prices. *The Journal of Finance*, 53(2):499–547, 1998.

[3] C. Alexander, A. Kaeck, and L. M. Nogueira. Model risk adjusted hedge ratios. *Journal of Futures Markets*, 29(11):1021–1049, 2009.

[4] Leif Andersen and Rupert Brotherton-Ratcliffe. The equity option volatility smile: an implicit finite-difference approach. *The Journal of Computational Finance*, 1(2):5–32, 1998.

[5] Leif Andersen and Rupert Brotherton-Ratcliffe. The equity option volatility smile: an implicit finite-difference approach. *The Journal of Computational Finance*, 1(2):5–32, 1998.

[6] Jesper Andreasen and Brian Norsk Huge. Volatility interpolation. *Available at SSRN 1694972*, 2010.

[7] F. Angelini and S. Herzel. Measuring the error of dynamic hedging: a laplace transform approach. *Journal of Computational Finance*, 13:47–72, 2009.

[8] F. Angelini and S. Herzel. Explicit formulas for the minimal hedging strategy in a martingale case. *Decisions in Economics and Finance*, 33:63–79, 2010.

[9] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[10] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[11] Gurdip Bakshi, Charles Cao, and Zhiwu Chen. *The Journal of finance*, 52(5):2003–2049, 1997.

[12] Gurdip Bakshi, Charles Cao, and Zhiwu Chen. *The Journal of finance*, 52(5):2003–2049, 1997.

[13] Gurdip Bakshi, Charles Cao, and Zhiwu Chen. Empirical performance of alternative option pricing models. *The Journal of finance*, 52(5):2003–2049, 1997.

[14] Gurdip Bakshi, Charles Cao, and Zhiwu Chen. *The Journal of finance*, 52(5):2003–2049, 1997.

[15] Gurdip Bakshi, Charles Cao, and Zhiwu Chen. *The Journal of finance*, 52(5):2003–2049, 1997.

[16] Bruce Bartlett. Hedging under sabr model. *Wilmott magazine*, 4:2–4, 2006.

[17] Julia Bennell and Charles Sutcliffe. Black–scholes versus artificial neural networks in pricing ftse 100 options. *Intelligent Systems in Accounting, Finance and Management*, 12 (4):243–260, 2004.

[18] Maxime Bergeron, Nicholas Fung, Zissis Poulos, John C Hull, and Andreas Veneris. Variational autoencoders: A hands-off approach to volatility. *Available at SSRN 3827447*, 2021.

[19] Fischer Black. The pricing of commodity contracts. *Journal of financial economics*, 3 (1-2):167–179, 1976.

[20] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *The journal of political economy*, pages 637–654, 1973.

[21] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[22] Tim Bollerslev, Julia Litvinova, and George Tauchen. Leverage and volatility feedback effects in high-frequency data. *Journal of Financial Econometrics*, 4(3):353–384, 2006.

[23] Matthew Brand. Fast online svd revisions for lightweight recommender systems. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 37–46. SIAM, 2003.

[24] Douglas T Breeden and Robert H Litzenberger. Prices of state-contingent claims implicit in option prices. *Journal of business*, pages 621–651, 1978.

[25] Mark Broadie, Jérôme Detemple, Eric Ghysels, and Olivier Torrés. American options with stochastic dividends and volatility: A nonparametric investigation. *Journal of Econometrics*, 94(1):53–92, 2000.

[26] Mark Broadie, Jérôme Detemple, Eric Ghysels, and Olivier Torrès. Nonparametric estimation of american options' exercise boundaries and call prices. *Journal of Economic Dynamics and Control*, 24(11):1829–1857, 2000.

[27] Bernhard Brunner and Reinhold Hafner. Arbitrage-free estimation of the risk-neutral density from the implied volatility smile. *Journal of Computational Finance*, 7(1):75–106, 2003.

[28] Hans Buehler, Lukas Gonon, Ben Wood, Josef Teichmann, Baranidharan Mohan, and Jonathan Kochems. Deep hedging: Hedging derivatives under generic market frictions using reinforcement learning-machine learning version. *Available at SSRN*, 2019.

[29] Richard H Byrd, Robert B Schnabel, and Gerald A Shultz. A trust region algorithm for nonlinearly constrained optimization. *SIAM Journal on Numerical Analysis*, 24(5):1152–1170, 1987.

[30] Yi Cao, Xiaoquan Liu, and Jia Zhai. Option valuation under no-arbitrage constraints with neural networks. *European Journal of Operational Research*, 293(1):361–374, 2021.

[31] P. Carr. European put call symmetry. pages 509–537. Cornell University working paper, 1994.

[32] P. Carr, K. Ellis, and V. Gupta. Static hedging of exotic options. *Journal of Finance*, 53: 1165–1190, 1998.

[33] Peter Carr and Dilip B Madan. A note on sufficient conditions for no arbitrage. *Finance Research Letters*, 2(3):125–130, 2005.

[34] Don M Chance, Thomas A Hanson, Weiping Li, and Jayaram Muthuswamy. A bias in the volatility smile. *Review of Derivatives Research*, 20(1):47–90, 2017.

[35] Bin Chen, Lech A Grzelak, and Cornelis W Oosterlee. Calibration and monte carlo pricing of the sabr-hull-white model for long-maturity equity derivatives. *The Journal of Computational Finance (79–113) Volume*, 15, 2011.

[36] Xiaohong Chen. Large sample sieve estimation of semi-nonparametric models. *Handbook of econometrics*, 6:5549–5632, 2007.

[37] Xiaohong Chen, Jeffrey Racine, and Norman R Swanson. Semiparametric arx neural-network models with an application to forecasting inflation. *IEEE Transactions on neural networks*, 12(4):674–683, 2001.

[38] Xilun Chen and K Selcuk Candan. Lwi-svd: low-rank, windowed, incremental singular value decompositions on time-evolving data sets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 987–996. ACM, 2014.

[39] Shunfeng Cheng and Michael Pecht. Using cross-validation for model parameter selection of sequential probability ratio test. *Expert Systems with Applications*, 39(9):8467–8473, 2012.

[40] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[41] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17 (11):1875–1886, 2015.

[42] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[43] T. F. Coleman, Y. Li, and M. Patron. Total risk minimization using Monte-carlo simulations. In John R. Birge and Vadim Linetsky, editors, *Handbook of Financial Engineering*. Elsevier, 2007.

[44] Thomas F. Coleman, Yuying Li, and Arun Verma. Reconstructing the unknown local volatility function. *The Journal of Computational Finance*, 2(3):77–102, 1999.

[45] Thomas F Coleman, Yohan Kim, Yuying Li, and Arun Verma. Dynamic hedging with a deterministic local volatility function model. *The Journal of Risk*, 5(6):63–89, 2001.

[46] Thomas F Coleman, Yuying Li, and Arun Verma. Reconstructing the unknown local volatility function. In *Quantitative Analysis In Financial Markets: Collected Papers of the New York University Mathematical Finance Seminar (Volume II)*, pages 192–215. World Scientific, 2001.

[47] Thomas F. Coleman, Yuying Li, and M. Patron. Discrete hedging under piecewise linear risk minimization. *The Journal of Risk*, 5:39–65, 2003.

[48] Thomas F Coleman, Yuying Li, and Maria-Cristina Patron. Discrete hedging under piecewise linear risk-minimization. *Option Pricing, Interest Rates and Risk Management*, 5: 39–65, 2003.

[49] Thomas F Coleman, Dmitriy Levchenkov, and Yuying Li. Discrete hedging of american-type options using local risk minimization. *Journal of Banking & Finance*, 31(11):3398–3419, 2007.

[50] Thomas F Coleman, Yuying Li, and Maria-Cristina Patron. Total risk minimization using monte carlo simulations. *Handbooks in Operations Research and Management Science*, 15:593–635, 2007.

[51] Ronan Collobert, Fabian Sinz, Jason Weston, and Léon Bottou. Trading convexity for scalability. In *Proceedings of the 23rd international conference on Machine learning*, pages 201–208. ACM, 2006.

[52] Tim Cooijmans, Nicolas Ballas, César Laurent, Çağlar Gülçehre, and Aaron Courville. Recurrent batch normalization. *arXiv preprint arXiv:1603.09025*, 2016.

[53] Jacopo Corbetta, Pierre Cohort, Ismail Laachir, and Claude Martini. Robust calibration and arbitrage-free interpolation of ssvi slices. *Decisions in Economics and Finance*, 42 (2):665–677, 2019.

[54] Corinna Cortes and Vladimir Vapnik. Support vector machine. *Machine learning*, 20(3): 273–297, 1995.

[55] John C Cox, Jonathan E Ingersoll Jr, and Stephen A Ross. A theory of the term structure of interest rates. In *Theory of valuation*, pages 129–164. World Scientific, 2005.

[56] S. Crépey. Delta-hedging vega risk. *Quantitative Finance*, 4(October):559–579, 2004.

[57] Stéphane Crépey. Delta-hedging vega risk? *Quantitative Finance*, 4(5):559–579, 2004.

[58] Toby Daglish, John Hull, and Wulin Suo. Volatility surfaces: theory, rules of thumb, and empirical evidence. *Quantitative Finance*, 7(5):507–524, 2007.

[59] MAH Dempster and Darren G Richards. Pricing american options fitting the smile. *Mathematical Finance*, 10(2):157–177, 2000.

[60] E. Derman and I. Kani. Riding on a smile. *Risk*, 7:32–39, 1994.

[61] E. Derman, I. Kani, and J. Zou. The local volatility surface: Unlocking the information in index option prices. *Financial Analysts Journal*, pages 25–36, 1996.

[62] J Duan, Genevieve Gauthier, J Simonato, and Caroline Sasseville. Approximating the gjr-garch and egarch option pricing models analytically. *Journal of Computational Finance*, 9(3):41, 2006.

[63] Jin-Chuan Duan. The garch option pricing model. *Mathematical finance*, 5(1):13–32, 1995.

[64] Jin-Chuan Duan, Genevieve Gauthier, and Jean-Guy Simonato. *An analytical approximation for the GARCH option pricing model*. École des hautes études commerciales, Groupe de recherche en finance, 1997.

[65] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul): 2121–2159, 2011.

[66] Bernard Dumas, Jeff Fleming, and Robert E Whaley. Implied volatility functions: Empirical tests. *The Journal of Finance*, 53(6):2059–2106, 1998.

[67] B. Dupire. Pricing with a smile. *Risk*, 7:18–20, 1994.

[68] Bruno Dupire et al. Pricing with a smile. *Risk*, 7(1):18–20, 1994.

[69] Nicole El Karoui and Marie-Claire Quenez. Dynamic programming and pricing of contingent claims in an incomplete market. *SIAM journal on Control and Optimization*, 33 (1):29–66, 1995.

[70] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[71] Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the Econometric Society*, pages 987–1007, 1982.

[72] Marcelo Espinoza, Johan AK Suykens, and Bart De Moor. Fixed-size least squares support vector machines: A large scale application in electrical load forecasting. *Computational Management Science*, 3(2):113–129, 2006.

[73] Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. Regularization networks and support vector machines. *Advances in computational mathematics*, 13(1):1–50, 2000.

[74] Yunlong Feng, Yuning Yang, Xiaolin Huang, Siamak Mehrkanoon, and Johan AK Suykens. Robust support vector machines for classification with nonconvex and smooth losses. *Neural computation*, 2016.

[75] Matthias R Fengler. Arbitrage-free smoothing of the implied volatility surface. *Quantitative Finance*, 9(4):417–428, 2009.

[76] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[77] H. Föllmer and M. Schweizer. Hedging by sequential regression: An introduction to the mathematics of option trading. *The ASTIN Bulletin*, 1:147–160, 1989.

[78] Hans Föllmer and Peter Leukert. Quantile hedging. *Finance and Stochastics*, 3(3):251–273, 1999.

[79] Hans Föllmer and Peter Leukert. Efficient hedging: cost versus shortfall risk. *Finance and Stochastics*, 4(2):117–146, 2000.

[80] Kenneth R French, G William Schwert, and Robert F Stambaugh. Expected stock returns and volatility. *Journal of financial Economics*, 19(1):3–29, 1987.

[81] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[82] René Garcia and Ramazan Gençay. Pricing and hedging derivative securities with neural networks and a homogeneity hint. *Journal of Econometrics*, 94(1):93–115, 2000.

[83] Jim Gatheral. A parsimonious arbitrage-free implied volatility parameterization with application to the valuation of volatility derivatives. *Presentation at Global Derivatives & Risk Management, Madrid*, page 0, 2004.

[84] Jim Gatheral. *The volatility surface: a practitioner's guide*, volume 357. John Wiley & Sons, 2011.

[85] Jim Gatheral and Antoine Jacquier. Arbitrage-free svi volatility surfaces. *Quantitative Finance*, 14(1):59–71, 2014.

[86] Ramazan Gençay and Min Qi. Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging. *IEEE Transactions on Neural Networks*, 12(4):726–734, 2001.

[87] Ramazan Gençay, Aslihan Salih, et al. Degree of mispricing with the black-scholes model and nonparametric cures. *Annals of Economics and Finance*, 4:73–102, 2003.

[88] Federico Girosi, Michael Jones, and Tomaso Poggio. Regularization theory and neural networks architectures. *Neural computation*, 7(2):219–269, 1995.

[89] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[90] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[91] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[92] S. Goutte, N. Oudjane, and F. Russo. Variance optimal hedging for discrete time processes with independent increments, application to electricity markets. *Journal of Computational Finance*, 17:71–111, 2013.

[93] Nikola Gradojevic, Ramazan Gençay, and Dragan Kukolj. Option pricing with modular neural networks. *IEEE transactions on neural networks*, 20(4):626–637, 2009.

[94] Ming Gu and Stanley C Eisenstat. A stable and fast algorithm for updating the singular value decomposition, 1994.

[95] Shuxin Guo and Qiang Liu. The black-scholes-merton dual equation. *Available at SSRN 3160399*, 2018.

[96] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[97] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

[98] Patrick Hagan and Andrew Lesniewski. Bartlett's delta in the sabr model. *Available at SSRN 2950749*, 2017.

[99] Patrick S Hagan, Deep Kumar, Andrew S Lesniewski, and Diana E Woodward. Managing smile risk. *The Best of Wilmott*, page 249, 2002.

[100] C. He, J.S. Kennedy, T. F. Coleman, P. A. Forsyth, Y. Li, and K. R. Vetzal. Calibration and hedging under jump diffusion. *Review of Derivative Research*, 9(1):1–35, 2006.

[101] D. Heath, E. Platen, and M. Schweizer. Numerical comparison of local risk-minimisation and mean-variance hedging. In *Option pricing, interest rates and risk management*, pages 509–537. (ed. E. Jouini, J. Cvitanic and, M. Musiela), Cambridge Univ. Press, 2001b.

[102] Sebas Hendriks and Claude Martini. The extended ssvi volatility surface. *Available at SSRN 2971502*, 2017.

[103] Steven L Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of financial studies*, 6(2):327–343, 1993.

[104] Steven L Heston and Saikat Nandi. A closed-form garch option valuation model. *The review of financial studies*, 13(3):585–625, 2000.

[105] Morris W Hirsch. Convergent activation dynamics in continuous time networks. *Neural networks*, 2(5):331–349, 1989.

[106] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[107] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[108] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[109] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[110] Peter J Huber et al. Robust estimation of a location parameter. *The annals of mathematical statistics*, 35(1):73–101, 1964.

[111] John Hull and Alan White. The pricing of options on assets with stochastic volatilities. *The journal of finance*, 42(2):281–300, 1987.

[112] John Hull and Alan White. Optimal delta hedging for options. *Journal of Banking & Finance*, 82:180–190, 2017.

[113] John C Hull. *Options, futures, and other derivatives*. Pearson Education India, 2006.

[114] James M Hutchinson, Andrew W Lo, and Tomaso Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, 49(3):851–889, 1994.

[115] Andrey Itkin. Deep learning calibration of option pricing models: some pitfalls and solutions. *arXiv preprint arXiv:1906.03507*, 2019.

[116] Jens Jackwerth and Mark Rubinstein. Recovering probability distributions from option prices. *The Journal of Finance*, 51(5):1611–1631, 1996.

[117] Ling Jian, Zhonghang Xia, Xijun Liang, and Chuanhou Gao. Design of a multiple kernel learning algorithm for ls-svm by convex programming. *Neural Networks*, 24(5):476–483, 2011.

[118] Licheng Jiao, Liefeng Bo, and Ling Wang. Fast sparse approximation for least squares support vector machine. *IEEE Transactions on Neural Networks*, 18(3):685–697, 2007.

[119] Michael I Jordan. Serial order: A parallel distributed processing approach. Technical report, CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE, 1986.

[120] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[121] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

[123] Nathan Lassance and Frédéric Vrins. A comparison of pricing and hedging performances of equity derivatives models. *Applied Economics*, 50(10):1122–1137, 2018.

[124] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[125] Felix Lenders, Christian Kirches, and Andreas Potschka. trlib: A vector-free implementation of the gltr method for iterative solution of the trust region problem. *Optimization Methods and Software*, 33(3):420–449, 2018.

[126] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[127] Shuaiqiang Liu, Cornelis W Oosterlee, and Sander M Bohte. Pricing options and computing implied volatilities using neural networks. *Risks*, 7(1):16, 2019.

[128] Philip M Long and Rocco A Servedio. Random classification noise defeats all convex potential boosters. *Machine learning*, 78(3):287–304, 2010.

[129] Andrew L Maas, Quoc V Le, Tyler M O'Neil, Oriol Vinyals, Patrick Nguyen, and Andrew Y Ng. Recurrent neural networks for noise reduction in robust asr. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[130] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.

[131] Mary Malliaris and Linda Salchenberger. A neural network model for estimating option prices. *Applied Intelligence*, 3(3):193–206, 1993.

[132] Benoit Mandelbrot. The variation of certain speculative prices. *The journal of business*, 36(4):394–419, 1963.

[133] Elena Marchiori, Niels HH Heegaard, Mikkel West-Nielsen, and Connie R Jimenez. Feature selection for classification with proteomic data of mixed quality. In *Computational Intelligence in Bioinformatics and Computational Biology, 2005. CIBCB'05. Proceedings of the 2005 IEEE Symposium on*, pages 1–7. IEEE, 2005.

[134] James Martens. Deep learning via hessian-free optimization. In *ICML*, volume 27, pages 735–742, 2010.

[135] James Martens and Ilya Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1033–1040. Citeseer, 2011.

[136] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[137] James Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209:415–446, 1909.

[138] R. Merton. The theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4:141–183, 1973.

[139] Robert C Merton. Theory of rational option pricing. *The Bell Journal of economics and management science*, pages 141–183, 1973.

[140] Dmytro Mishkin and Jiri Matas. All you need is a good init. *arXiv preprint arXiv:1511.06422*, 2015.

[141] Michael C Mozer. A focused back-propagation algorithm for temporal pattern recognition. *Complex systems*, 3(4):349–381, 1989.

[142] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence o (1/k2). In *Doklady an SSSR*, volume 269, pages 543–547, 1983.

[143] Ke Nian, Thomas F Coleman, and Yuying Li. Learning minimum variance discrete hedging directly from the market. *Quantitative Finance*, pages 1–14, 2018.

[144] Ke Nian, Thomas F Coleman, and Yuying Li. Learning sequential option hedging models from market data. *Journal of Banking and Finance*, 2019. Accepted Pending Revision.

[145] Ke Nian, Thomas F Coleman, and Yuying Li. Learning sequential total hedging models from market data. Submitted, 2019.

[146] Christopher Olah. Understanding lstm networks. *GITHUB blog, posted on August*, 27: 2015, 2015.

[147] LLC OptionMetrics. Ivy db file and data reference manual, 2008.

[148] Tapio Pahikkala, Jorma Boberg, and Tapio Salakoski. Fast n-fold cross-validation for regularized least-squares. In *Proceedings of the ninth Scandinavian conference on artificial intelligence (SCAI 2006)*, pages 83–90. Otamedia Oy, Espoo, Finland, 2006.

[149] Fernando De Meer Pardo and Rafael Cobo López. Mitigating overfitting on financial datasets with generative adversarial networks. *The Journal of Financial Data Science*, 2 (1):76–85, 2020.

[150] Robert J Plemmons. M-matrix characterizations. i—nonsingular m-matrices. *Linear Algebra and its applications*, 18(2):175–188, 1977.

[151] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[152] K. Poulsen R., R. Schenk-Hoppé, and C.-O Ewald. Risk minimization in stochastic volatility models: model risk and empirical performance. *Quantitative Finance*, 9(6):693–704, 2009.

[153] Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *The Journal of Machine Learning Research*, 15(1):335–366, 2014.

[154] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[155] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[156] Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, 53(1-2):23–69, 2003.

[157] LCG Rogers and MR Tehranchi. Can the implied volatility surface move by parallel shifts? *Finance and Stochastics*, 14(2):235–248, 2010.

[158] Mark Rubinstein. Implied binomial trees. *The Journal of Finance*, 49(3):771–818, 1994.

[159] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[160] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[161] M. Schäl. On quadratic cost criteria for option hedging. *Mathematics of Operation Research*, 19(1):121–131, 1994.

[162] M. Schweizer. Variance-optimal hedging in discrete time. *Mathematics of Operation Research*, 20:1–32, 1995.

[163] Martin Schweizer. A guided tour through quadratic hedging approaches. *Option Pricing, Interest Rates and Risk Management*, page 538.

[164] Martin Schweizer. Variance-optimal hedging in discrete time. *Mathematics of Operations Research*, 20(1):1–32, 1995.

[165] Steven E Shreve. *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer Science & Business Media, 2004.

[166] Alex J Smola and Bernhard Schökopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 911–918. Morgan Kaufmann Publishers Inc., 2000.

[167] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[168] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[169] Richard S Sutton. Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proc. 8th annual conf. cognitive science society*, pages 823–831. Erlbaum, 1986.

[170] Johan AK Suykens, Tony Van Gestel, and Jos De Brabanter. *Least squares support vector machines*. World Scientific, 2002.

[171] Mingkui Tan, Li Wang, and Ivor W Tsang. Learning sparse svm for feature selection on very high dimensional datasets. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 1047–1054, 2010.

[172] Aditya Tayal, Thomas F Coleman, and Yuying Li. Primal explicit max margin feature selection for nonlinear support vector machines. *Pattern Recognition*, 47(6):2153–2164, 2014.

[173] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[174] Vladimir Naumovich Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[175] Manik Varma and Bodla Rakesh Babu. More generality in efficient multiple kernel learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1065–1072. ACM, 2009.

[176] Grace Wahba. *Spline models for observational data*, volume 59. Siam, 1990.

[177] Kuaini Wang and Ping Zhong. Robust non-convex least squares loss function for regression with outliers. *Knowledge-Based Systems*, 71:290–302, 2014.

[178] Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

[179] Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

[180] Yichao Wu and Yufeng Liu. Robust truncated hinge loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983, 2007.

[181] Linli Xu, Koby Crammer, and Dale Schuurmans. Robust support vector machine training via convex outlier ablation. In *AAAI*, volume 6, pages 536–542, 2006.

[182] Peng Xu, Farbod Roosta-Khorasan, and Michael W Mahoney. Second-order optimization for non-convex machine learning: An empirical study. *arXiv preprint arXiv:1708.07827*, 2017.

[183] Xiaowei Yang, Liangjun Tan, and Lifang He. A robust least squares support vector machine for regression and classification with noise. *Neurocomputing*, 140:41–52, 2014.

[184] Jingtao Yao, Yili Li, and Chew Lim Tan. Option price forecasting using neural networks. *Omega*, 28(4):455–466, 2000.

[185] Zhewei Yao, Peng Xu, Farbod Roosta-Khorasani, and Michael W Mahoney. Inexact non-convex newton-type methods. *arXiv preprint arXiv:1802.06925*, 2018.

[186] Adonis Yatchew and Wolfgang Härdle. Nonparametric state price density estimation using constrained least squares and the bootstrap. *Journal of Econometrics*, 133(2):579–599, 2006.

[187] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

[188] Yao-liang Yu, Özlem Aslan, and Dale Schuurmans. A polynomial-time form of robust regression. In *Advances in Neural Information Processing Systems*, pages 2483–2491, 2012.

[189] Yaoliang Yu, Xun Zheng, Micol Marchetti-Bowick, and Eric Xing. Minimizing nonconvex non-separable functions. In *Artificial Intelligence and Statistics*, pages 1107–1115, 2015.

[190] Haynes HM Yung and Hua Zhang. An empirical investigation of the garch option pricing model: hedging performance. *Journal of Futures Markets: Futures, Options, and Other Derivative Products*, 23(12):1191–1207, 2003.

[191] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[192] Yu Zheng, Yongxin Yang, and Bowei Chen. Incorporating prior financial domain knowledge into neural networks for implied volatility surface prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3968–3975, 2021.

Appendix A

Cross Validation of Kernel Regularized Network

For our data-driven approach based on kernel regularized network methond, we need to select an appropriate penalty λ_P to control the model complexity. Cross-validation (CV) is a commonly used method for the performance estimation and model selection for the learning algorithms. For example, the Leave-One-Out Cross-Validation (LOOCV) computes the output for each data instance using parameters trained on the remaining data instances. For the regularized kernel methods, we can compute the CV error efficiently without retraining the model in each CV round [148, 176].

Recall that, for the regularized pricing model, the minimization problem is

$$\min_{\widehat{\boldsymbol{\alpha}}} \left(\sum_{i=1}^{M} \left(V^{mkt}_{t_i,T_i,K_i} - \sum_{j=1}^{M} \widehat{\alpha}_j \mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j}, \mathbf{x}_{t_i}^{T_i,K_i}) \right)^2 + \lambda_P \sum_{i=1}^{M} \sum_{j=1}^{M} \widehat{\alpha}_i \widehat{\alpha}_j \mathscr{K}(\mathbf{x}_{t_j}^{T_j,K_j}, \mathbf{x}_{t_i}^{T_i,K_i}) \right) \tag{A.1}$$

Here $V^{mkt}_{t_i,T_i,K_i}$ is the market option value for time t_i, expiry T_i and strike K_i. The vector $\mathbf{x}_{t_j}^{T_j,K_j}$ is the vector of corresponding input features and λ_P is the penalty parameter for the regulation. Let $\mathbb{K}$ be the kernel matrix with $\mathbb{K}_{ij} = \mathscr{K}(\mathbf{x}_i, \mathbf{x}_j)$. Note $\mathbb{K}$ here denotes the kernel matrix which should not be confused with the strike K. Problem (A.1) can be rewritten in matrix form:

$$\min_{\widehat{\boldsymbol{\alpha}} \in \mathbb{R}^M} (\mathbb{K}\widehat{\boldsymbol{\alpha}} - \mathbb{V})^T (\mathbb{K}\widehat{\boldsymbol{\alpha}} - \mathbb{V}) + \lambda_P \widehat{\boldsymbol{\alpha}}^T \mathbb{K} \widehat{\boldsymbol{\alpha}} \tag{A.2}$$

with

$$\mathbb{V} = \{V^{mkt}_{t_1,T_1,K_1}, \ldots, V^{mkt}_{t_M,T_M,K_M}\} \quad , \quad \widehat{\boldsymbol{\alpha}} = \{\widehat{\alpha}_1, \ldots, \widehat{\alpha}_M\}$$

The solution to (A.2) is:

$$\widehat{\boldsymbol{\alpha}}^* = (\mathbb{K} + \lambda_P \boldsymbol{I})^{-1} \mathbb{V}$$

Given the eigen-decomposition $\mathbb{K} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$, we can easily see that:

$$(\mathbb{K} + \lambda_P \boldsymbol{I})^{-1} = \boldsymbol{Q}(\boldsymbol{\Lambda} + \lambda_P \boldsymbol{I})^{-1}\boldsymbol{Q}^T \tag{A.3}$$

Because $(\boldsymbol{\Lambda} + \lambda_P I)$ is a diagonal matrix, given the eigen-decomposition $\mathbb{K} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$, we can get different solutions to (A.2) as λ_P varies in $O(M^2)$.

Let $V(\mathbf{x}_{t_j}^{T_j,K_j}; \widehat{\boldsymbol{\alpha}}^*)$ be the output for data instance j when regularized kernel methods (A.1) is trained on all training data instances. Let $V^l(\mathbf{x}_{t_j}^{T_j,K_j}; \widehat{\boldsymbol{\alpha}}^l)$ be the output for data instance j when regularized kernel methods (A.1) is trained on all training data instances except $\mathbf{x}_{t_l}^{T_l,K_l}$. Let $\mathbb{V}^l = \{\mathbb{V}_1^l, \mathbb{V}_2^l, \cdots, \mathbb{V}_M^l\}$ be the vector where $\mathbb{V}_j^l = V_{t_j,T_j,K_j}^{mkt}$ for $j \neq l$ and $\mathbb{V}_l^l = V^l(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^l)$. Since $V^l(\cdot; \widehat{\boldsymbol{\alpha}}^l)$ is the model trained on all example except for $\mathbf{x}_{t_l}^{T_l,K_l}$, it is easy to see that $V^l(\cdot; \widehat{\boldsymbol{\alpha}}^l)$ minimizes

$$\min_{\widehat{\boldsymbol{\alpha}} \in \mathbb{R}^M} (\mathbb{K}\widehat{\boldsymbol{\alpha}} - \mathbb{V}^l)^T(\mathbb{K}\widehat{\boldsymbol{\alpha}} - \mathbb{V}^l) + \lambda_P \widehat{\boldsymbol{\alpha}}^T \mathbb{K}\widehat{\boldsymbol{\alpha}} \tag{A.4}$$

The solution to (A.4) is:

$$\widehat{\boldsymbol{\alpha}}^l = (\mathbb{K} + \lambda_P \boldsymbol{I})^{-1}\mathbb{V}^l$$

Let $\boldsymbol{B} = \mathbb{K}(\mathbb{K} + \lambda_P \boldsymbol{I})^{-1}$, therefore, we have

$$\boldsymbol{B}\mathbb{V} = \{V(\mathbf{x}_{t_1}^{T_1,K_1}; \widehat{\boldsymbol{\alpha}}^*), \ldots, V(\mathbf{x}_{t_M}^{T_M,K_M}; \widehat{\boldsymbol{\alpha}}^*)\}$$

where $V(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*)$ is the option value funtion with the solution $\widehat{\boldsymbol{\alpha}}^*$ given by $\widehat{\boldsymbol{\alpha}}^* = (\mathbb{K} + \lambda_P \boldsymbol{I})^{-1}\mathbb{V}$. Let B_{ij} be the element of $\boldsymbol{B}$ of ith row and jth column. We can easily show that:

$$V^l(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^l) - V(\mathbf{x}_t^{T,K}; \widehat{\boldsymbol{\alpha}}^*) = \sum_{i=1}^{M} B_{li}(\mathbb{V}_i^l - \mathbb{V}_i) = B_{ll}(V^l(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^l) - \mathbb{V}_l) = B_{ll}(V^l(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^l) - V_{t_l,T_l,K_l}^{mkt})$$

Thus,

$$V^l(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^l) = \frac{V(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^*) - B_{ll}V_{t_l,T_l,K_l}^{mkt}}{1 - B_{ll}} \tag{A.5}$$

Therefore, we can get the $V^l(\mathbf{x}_{t_l}^{T_l,K_l}; \widehat{\boldsymbol{\alpha}}^l)$ without actually retraining the model.

The leave-one-out estimations are:

$$\mathbb{V}_{loo} = (I - \boldsymbol{B}_{LL})^{-1}(\boldsymbol{B}\mathbb{V} - \boldsymbol{B}_{LL}\mathbb{V}) \tag{A.6}$$

and the leave-one-out errors for all data instance are:

$$\mathbb{V} - \mathbb{V}_{loo} = \mathbb{V} - (\boldsymbol{I} - \boldsymbol{B}_{LL})^{-1}(\boldsymbol{B}\mathbb{V} - \boldsymbol{B}_{LL}\mathbb{V}) = (\boldsymbol{I} - \boldsymbol{B}_{LL})^{-1}(\mathbb{V} - \boldsymbol{B}\mathbb{V}) \tag{A.7}$$

where $\boldsymbol{B}_{LL}$ is a diagonal matrix with diagonal element of the matrix $\boldsymbol{B}$, i.e., B_{ii}, $i = 1, \ldots, M$, on its diagonal.

We can further simplify the expression by noting the fact that:

$$\begin{aligned}\boldsymbol{B} &= \mathbb{K}(\mathbb{K}+\lambda_P\boldsymbol{I})^{-1} \\ &= \boldsymbol{Q}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\lambda_P\boldsymbol{I})^{-1}\boldsymbol{Q}^T \\ &= \boldsymbol{Q}(\boldsymbol{\Lambda}+\lambda_P\boldsymbol{I}-\lambda_P\boldsymbol{I})(\boldsymbol{\Lambda}+\lambda_P\boldsymbol{I})^{-1}\boldsymbol{Q}^T \\ &= \boldsymbol{I}-\lambda_P\boldsymbol{Q}(\boldsymbol{\Lambda}+\lambda_P\boldsymbol{I})^{-1}\boldsymbol{Q}^T = \boldsymbol{I}-\lambda_P(\mathbb{K}+\lambda_P\boldsymbol{I})^{-1}\end{aligned}$$

Therefore, we get:

$$\begin{aligned}(\boldsymbol{I}-\boldsymbol{B}_{LL}) &= diag(\boldsymbol{I}-\boldsymbol{B}) = \lambda_P\, diag((\mathbb{K}+\lambda_P\boldsymbol{I})^{-1}) \\ \mathbb{V}-B\mathbb{V} &= \lambda_P(\mathbb{K}+\lambda_P\boldsymbol{I})^{-1}\mathbb{V} = \lambda_P\widehat{\boldsymbol{\alpha}}^*\end{aligned} \tag{A.8}$$

From equation (A.7) and (A.8) , we can get:

$$\mathbb{V}-\mathbb{V}_{loo} = diag((\mathbb{K}+\lambda_P\boldsymbol{I})^{-1})^{-1}\widehat{\boldsymbol{\alpha}}^* \tag{A.9}$$

Given the eigen-decomposition $\mathbb{K}=\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$, we can compute $\widehat{\boldsymbol{\alpha}}^*$ as λ_P varies in $O(M^2)$. Similarly, given the eigen-decomposition $\mathbb{K}=\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$, we can compute the diagonal of $(\mathbb{K}+\lambda_P\boldsymbol{I})^{-1}$ in $O(M^2)$. Then using (A.9), the LOOCV errors can be computed in $O(M^2)$. It can been further shown [148] that, given the eigen-decomposition $\mathbb{K}=\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$, the computational complexity for the n-fold cross-validation (nFCV) is $O(M^3/n)$. Interested readers can referred to [148] for more details.

Let $\Delta\mathbb{V}=\{\Delta V^{mkt}_{t_1,T_1,K_1},\cdots,\Delta V^{mkt}_{t_M,T_M,K_M}\}$ and let $\boldsymbol{D}$ be a diagonal matrix with $D_{ii}=\Delta S_{t_i}$. $i=1,\cdots,M$. on its diagonal. Similarly, we can rewrite the minimization problem in matrix form:

$$\min_{\widehat{\boldsymbol{\alpha}}\in\mathbb{R}^m}(\boldsymbol{D}\mathbb{K}\widehat{\boldsymbol{\alpha}}-\Delta\mathbb{V})^T(\boldsymbol{D}\mathbb{K}\widehat{\boldsymbol{\alpha}}-\Delta\mathbb{V})+\lambda_P\widehat{\boldsymbol{\alpha}}^T\mathbb{K}\widehat{\boldsymbol{\alpha}} \tag{A.10}$$

Let $\widetilde{\mathbb{K}}=\boldsymbol{D}\mathbb{K}$, the solution to (A.10) can be obtained by:

$$\widehat{\boldsymbol{\alpha}}^*=(\widetilde{\mathbb{K}}^T\widetilde{\mathbb{K}}+\lambda_P\mathbb{K})^{-1}\widetilde{\mathbb{K}}^T\Delta\mathbb{V}$$

In this thesis, we change the penalty term for (A.10) from $\lambda_P\widehat{\boldsymbol{\alpha}}^T\mathbb{K}\widehat{\boldsymbol{\alpha}}$ to $\lambda_P\widehat{\boldsymbol{\alpha}}^T\widehat{\boldsymbol{\alpha}}$, the minimization problem becomes:

$$\min_{\widehat{\boldsymbol{\alpha}}\in\mathbb{R}^m}(\boldsymbol{D}\mathbb{K}\widehat{\boldsymbol{\alpha}}-\Delta\check{\boldsymbol{V}})^T(\boldsymbol{D}\mathbb{K}\widehat{\boldsymbol{\alpha}}-\Delta\check{\boldsymbol{V}})+\lambda_P\widehat{\boldsymbol{\alpha}}^T\widehat{\boldsymbol{\alpha}} \tag{A.11}$$

The solution to (A.11) can be obtained by:

$$\widehat{\boldsymbol{\alpha}}^*=(\widetilde{\mathbb{K}}^T\widetilde{\mathbb{K}}+\lambda_P\boldsymbol{I})^{-1}\widetilde{\mathbb{K}}^T\Delta\mathbb{V}$$

Utilizing the ideas from [148, 176], we can similarly show that, given the singular value decomposition $\widetilde{\mathbb{K}}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, for the problem (A.11), the computational complexity for the LOOCV is

still $O(M^2)$ and the computational complexity for the nFCV is still $O(M^3/n)$. In practice, changing the penalty term from $\lambda_P \widehat{\boldsymbol{\alpha}}^T \mathbb{K} \widehat{\boldsymbol{\alpha}}$ to $\lambda_P \widehat{\boldsymbol{\alpha}}^T \widehat{\boldsymbol{\alpha}}$ will not affect the actual performance too much. Therefore, in order to improve the computation efficiency for the direct data-driven approach, we are solving the problem (A.11).

Appendix B

Fixing the Butterfly Arbitrage for SABR Model

In this thesis, we assume the adjusted p.d.f is of the following simplified form:

$$\hat{g}(x) = \begin{cases} \lambda_L \, q(x;\mu_L,\sigma_L), & \text{if } 0 < x < K_L \\ p(x;T,S_t), & \text{if } K_L \leq x \leq K_U \\ \lambda_U \, q(x;\mu_U,\sigma_U), & \text{if } x > K_U \end{cases} \tag{B.1}$$

Here, K_L and K_U are the lower and upper strike limits, within which the implicit probability density function (p.d.f) $p(x;T,S_t)$ from SABR model is assumed to be valid. The $q(x;\mu,\sigma)$ is the p.d.f of a log-normal distribution parametrized by μ and σ:

$$q(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}.$$

We require the following conditions to be satisfied:

1. Intergrability constraint

$$\int_0^{K_L} \lambda_L q(x;\mu_L,\sigma_L)dx + \int_{K_L}^{K_U} p(x;T,S_t)dx + \int_{K_U}^{\infty} \lambda_U \, q(x;\mu_U,\sigma_U)dx = 1 \tag{B.2}$$

2. Martingale constraint

$$\int_0^{K_L} x\lambda_L q(x;\mu_L,\sigma_L)dx + \int_{K_L}^{K_U} xp(x;T,S_t)dx + \int_{K_U}^{\infty} x\lambda_U \, q(x;\mu_U,\sigma_U)dx = F(t,T) \tag{B.3}$$

The intergrability constraint ensures that $\hat{g}(x)$ is a valid p.d.f. The martingale constraint ensures that $E[S_T] = F(t,T) = S_t e^{(r-q)(T-t)}$ under ther adjusted p.d.f.

Furthermore, we set $\mu_L = \ln(S_t) + (r - q - \frac{\sigma_L^2}{2})(T-t)$ and $\mu_U = \ln(S_t) + (r - q - \frac{\sigma_U^2}{2})(T-t)$ such that we have only four parameters to be solved: $\{\sigma_L, \sigma_U, \lambda_L, \lambda_U\}$.

Let:

- $C_{BS}(T,K;\sigma)$ and $P_{BS}(T,K;\sigma)$ be the call and put option prices from Black-Scholes model with the Black-Scholes volatility σ.
- $DC_{BS}(T,K;\sigma)$ and $DP_{BS}(T,K;\sigma)$ be the digital call and put option prices from Black-Scholes model with the Black-Scholes volatility σ where a digital call pays one dollar if the underlying price exceeds the strike and a digital put pays the same amount if the underlying is below the strike.
- σ_B be implied Black's volatility given by the SABR apprximation formula (2.2.11) for $K \in [K_L, K_U]$. For simplicity, we write $\sigma_B(K)$ to denote the implied Black's volatility given by the SABR formula (2.2.11) for a strike K since the other parameters in $\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho)$ remain unchanged in the following discussion for an expiry T at a time t.

By above settings, we will have:

$$D(t,T)\int_0^{K_L} \lambda_L q(x;\mu_L,\sigma_L)dx = \lambda_L DP_{BS}(T,K_L;\sigma_L) \tag{B.4}$$

$$D(t,T)\int_{K_U}^{\infty} \lambda_U\, q(x;\mu_U,\sigma_U)dx = \lambda_U DC_{BS}(T,K_U;\sigma_U) \tag{B.5}$$

$$D(t,T)\int_0^{K_L} (K_L - x)\lambda_L q(x;\mu_L,\sigma_L)dx = \lambda_L P_{BS}(T,K_L;\sigma_L) \tag{B.6}$$

$$D(t,T)\int_{K_U}^{\infty} (x - K_U)\lambda_U\, q(x;\mu_U,\sigma_U)dx = \lambda_U C_{BS}(T,K_U;\sigma_U) \tag{B.7}$$

If we have the below equations hold:

$$\lambda_L DP_{BS}(T,K_L;\sigma_L) = DP_{BS}(T,K_L;\sigma_B(K_L)) \tag{B.8}$$

$$\lambda_U DC_{BS}(T,K_U;\sigma_U) = DC_{BS}(T,K_U;\sigma_B(K_U)) \tag{B.9}$$

$$\lambda_L P_{BS}(T,K_L;\sigma_L) = P_{BS}(T,K_L;\sigma_B(K_L)) \tag{B.10}$$

$$\lambda_U C_{BS}(T,K_U;\sigma_U) = C_{BS}(T,K_U;\sigma_B(K_U)) \tag{B.11}$$

we can show the intergrability constraint (B.2) and martingale constraint (B.3) are also satisified. Furthermore, the solution to the equantions (B.8) to (B.11) is $\{\sigma_L = \sigma_B(K_L), \sigma_U = \sigma_B(K_U), \lambda_L = 1, \lambda_U = 1\}$. Therefore, if we have $\{\sigma_L = \sigma_B(K_L), \sigma_U = \sigma_B(K_U), \lambda_L = 1, \lambda_U = 1\}$, the intergrability constraint (B.2) and martingale constraint (B.3) are satisified.

Here, we firstly show the intergrability constraint (B.2) is satisified if we have equations (B.8) to (B.11) hold. Comnbining equation (B.4) and equation (B.8), we have:

$$\frac{1}{D(t,T)}DP_{BS}(T,K_L;\sigma_B(K_L)) = \frac{1}{D(t,T)}\lambda_L DP_{BS}(T,K_L;\sigma_L) = \int_0^{K_L}\lambda_L q(x;\mu_L,\sigma_L)dx \tag{B.12}$$

Similarly, comnbining equation (B.5) and equation (B.9), we have

$$\frac{1}{D(t,T)}DC_{BS}(T,K_U;\sigma_B(K_U)) = \frac{1}{D(t,T)}\lambda_L DC_{BS}(T,K_U;\sigma_U) = \int_{K_U}^{\infty}\lambda_L q(x;\mu_L,\sigma_L)dx \tag{B.13}$$

We further note that:

$$\begin{aligned}\int_{K_L}^{K_U} p(x;T,S_t) &= \int_{K_L}^{\infty} p(x;T,S_t) - \int_{K_U}^{\infty} p(x;T,S_t) \\ &= \frac{1}{D(t,T)}DC_{BS}(T,K_L;\sigma_B(K_L)) - \frac{1}{D(t,T)}DC_{BS}(T,K_U;\sigma_B(K_U))\end{aligned} \tag{B.14}$$

By call-put parity, we have:

$$DC_{BS}(T,K_L;\sigma_B(K_L)) = D(t,T) - DP_{BS}(T,K_L;\sigma_B(K_L))$$

Therefore, we have:

$$\begin{aligned}\int_{K_L}^{K_U} p(x;T,S_t) &= \int_{K_L}^{\infty} p(x;T,S_t) - \int_{K_U}^{\infty} p(x;T,S_t) \\ &= 1 - \frac{1}{D(t,T)}DP_{BS}(T,K_L;\sigma_B(K_L)) - \frac{1}{D(t,T)}DC_{BS}(T,K_U;\sigma_B(K_U))\end{aligned} \tag{B.15}$$

Combining equations (B.12) and (B.13) with equation (B.15), we show that the intergrability constraint (B.2) is satisfied.

In terms of the martingale constraint, we firstly split the equation (B.6) as follows:

$$\int_0^{K_L}(K_L - x)\lambda_L\, q(x;\mu_L,\sigma_L)dx = \int_0^{K_L} K\lambda_L\, q(x;\mu_L,\sigma_L)dx - \int_0^{K_L} x\lambda_L\, q(x;\mu_L,\sigma_L)dx \tag{B.16}$$

Plugging in equation (B.4), we have:

$$K_L\lambda_L DP_{BS}(T,K_L;\sigma_L) - \lambda_L P_{BS}(T,K_L;\sigma_L) = D(t,T)\int_0^{K_L} x\lambda_L\, q(x;\mu_L,\sigma_L)dx \tag{B.17}$$

Similary, we have:

$$K_U\lambda_U DC_{BS}(T,K_U;\sigma_U) + \lambda_U C_{BS}(T,K_U;\sigma_U) = D(t,T)\int_{K_U}^{\infty} x\lambda_U\, q(x;\mu_U,\sigma_U)dx \tag{B.18}$$

Furthermore, we can show that for $K_L < K < K_U$:

$$D(t,T)\int_{K_L}^{K_U}(x-K)p(x;T,S_t)dx = [C_{BS}(T,K_L;\sigma_B(K_L)) + (K_L-K)DC_{BS}(T,K_L;\sigma_B(K_L))] \\ -[C_{BS}(T,K_U;\sigma_B(K_U)) + (K_U-K)DC_{BS}(T,K_L;\sigma_B(K_U))] \tag{B.19}$$

From equation (B.15) we know that:

$$D(t,T)\int_{K_L}^{K_U}Kp(x;T,S_t)dx = K[D(t,T) - DP_{BS}(T,K_L;\sigma_B(K_L)) - DC_{BS}(T,K_U;\sigma_B(K_U))]$$

Therefore, we have

$$\begin{aligned} D(t,T)\int_{K_L}^{K_U}xp(x;T,S_t)dx = &[C_{BS}(T,K_L;\sigma_B(K_L)) + (K_L-K)DC_{BS}(T,K_L;;\sigma_B(K_L))] \\ &-[C_{BS}(T,K_U;\sigma_B(K_U)) + (K_U-K)DC_{BS}(T,K_L;\sigma_B(K_U))] \\ &+K[D(t,T) - DP_{BS}(T,K_L;\sigma_B(K_L)) - DC_{BS}(T,K_U;\sigma_B(K_U))] \end{aligned} \tag{B.20}$$

By the call-put parity, we have:

$$\begin{aligned} C_{BS}(T,K_L;\sigma_B(K_L)) &= D(t,T)[F(t,T)-K_L] + P_{BS}(T,K_L;\sigma_B(K_L)) \\ DC_{BS}(T,K_L;\sigma_B(K_L)) &= D(t,T) - DP_{BS}(T,K_L;\sigma_B(K_L)) \end{aligned} \tag{B.21}$$

Combining equation (B.20) and equations (B.21), we then have:

$$\begin{aligned} \int_{K_L}^{K_U}xp(x;T,S_t)dx = &F(t,T) - \frac{1}{D(t,T)}[K_L DP_{BS}(T,K_L;\sigma_B(K_L)) - P_{BS}(T,K_L;\sigma_B(K_L))] - \\ &\frac{1}{D(t,T)}[K_U DC_{BS}(T,K_U;\sigma_B(K_U)) + C_{BS}(T,K_U;\sigma_B(K_U))] \end{aligned} \tag{B.22}$$

Plugging in equations (B.8) to (B.11) into equation (B.22), we have

$$\begin{aligned} \int_{K_L}^{K_U}xp(x;T,S_t)dx = &F(t,T) - \frac{1}{D(t,T)}[K_L DP_{BS}(T,K_L;\sigma_L) - P_{BS}(T,K_L;\sigma_L)] - \\ &\frac{1}{D(t,T)}[K_U DC_{BS}(T,K_U;\sigma_U) + C_{BS}(T,K_U;\sigma_U)] \end{aligned} \tag{B.23}$$

Lastly, combining equation (B.16), (B.17) and (B.23) together,

$$\begin{aligned} F(t,T) =& \frac{1}{D(t,T)}\lambda_L[K_L DP_{BS}(T,K_L;\sigma_L) - P_{BS}(T,K_L;\sigma_L)] + \\ & F(t,T) - \frac{1}{D(t,T)}[K_L DP_{BS}(T,K_L;\sigma_L) - P_{BS}(T,K_L;\sigma_L)] - \\ & \frac{1}{D(t,T)}[K_U DC_{BS}(T,K_U;\sigma_U) + C_{BS}(T,K_U;\sigma_U)] \\ & \frac{1}{D(t,T)}\lambda_U[K_U DC_{BS}(T,K_U;\sigma_U) + C_{BS}(T,K_U;\sigma_U)] \end{aligned} \tag{B.24}$$

We therefore prove the martingale constraint (B.3) is satisified.

Since the adjusted p.d.f (B.1) satisfies intergrability constraint (B.2) and martingale constraint (B.3) with the setting:

$$\begin{aligned} & \sigma_L = \sigma_B(K_L)\ ,\ \sigma_U = \sigma_B(K_U) \\ & \lambda_L = 1\ ,\ \lambda_U = 1 \\ & \mu_L = \ln(S_t) + (r - q - \frac{\sigma_B(K_L)^2}{2})(T-t) \\ & \mu_U = \ln(S_t) + (r - q - \frac{\sigma_B(K_U)^2}{2})(T-t) \end{aligned} \tag{B.25}$$

and will not be negative at tails of the distribution because that the two tails are from two log-normal distribuions, we can conclude that the adjusted $\overline{C}_{SABR}(T,K)$ will be free of butterfly arbitrage.

In this thesis, we use the risk neutral adjustment (B.1) with 4 parameters $\{\sigma_L, \sigma_U, \lambda_L, \lambda_U\}$ due to its simplicity. Notice that density function $\hat{g}(x)$ calibrated following the above risk neutral adjustment maybe discontinuous at K_L and K_U. If one want to ensure the continuity of the density function, one can solve the overdetermined system suggested by Brunner and Hafner [27].

Appendix C

No-Arbitrage Properties of the Volatility Interpolation Algorithm

Define the normalized call price $\widehat{C}(T,\widehat{K})$ in terms of discount factor $D(t,T)$ and forward price $F(t,T)$ and the normalized strike $\widehat{K}$ as:

$$\begin{aligned} D(t,T) &= e^{-r(T-t)} \\ F(t,T) &= S_t e^{(r-q)(T-t)} \\ \widehat{K} &= \frac{K}{F(t,T)} \\ \widehat{C}(T,\widehat{K}) &= \frac{C(T,\widehat{K}F(t,T))}{D(t,T)F(t,T)} = \frac{C(T,K)}{D(t,T)F(t,T)} \end{aligned}$$

In this thesis, we use the Dupire's equation to interpolate the implied volatilities between expiries:

$$\frac{\partial \widehat{C}(T,\widehat{K})}{\partial T} = \frac{1}{2}\widehat{\sigma}^2(T,\widehat{K})\widehat{K}^2\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}, \;\; \widehat{\sigma}(T,\widehat{K}) = \sigma(T,K)$$

Assume we are given a grid of expiries available in market $t = T_0 < T_1 < \cdots < T_M$ and a grid of normalized strike: $0 = \widehat{K}_0 < \widehat{K}_1 < \cdots < \widehat{K}_N$, and $\widehat{\sigma}(T,\widehat{K})$ is a piecewise constant functions for a given T_i.

$$\widehat{\sigma}(T_i,\widehat{K}) = \begin{cases} \widehat{\sigma}_{T_i,\widehat{K}_0}, & \text{if } \widehat{K} \le \widehat{K}_0 \\ \vdots & \vdots \\ \widehat{\sigma}_{T_i,\widehat{K}_j}, & \text{if } \widehat{K}_{j-1} < \widehat{K} \le \widehat{K}_j \\ \vdots & \vdots \\ \widehat{\sigma}_{T_i,\widehat{K}_N}, & \text{if } \widehat{K} > \widehat{K}_N \end{cases} \tag{C.1}$$

The fully implicit finite difference method in matrix form is:

$$\begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix} = (I - \mathbb{S}\mathbb{D}(T_{i+1}-T_i)) \begin{bmatrix} \widehat{C}(T_{i+1},\widehat{K}_0) \\ \widehat{C}(T_{i+1},\widehat{K}_1) \\ \widehat{C}(T_{i+1},\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_{i+1},\widehat{K}_N) \end{bmatrix} \tag{C.2}$$

where I is the identity matrix, $\mathbb{S}$ is a diagonal matrix parameterized by $\widehat{\sigma}(T_i,.)$ as in equation (C.1), $\mathbb{D}$ is proportional to the discrete second order difference matrix and $(T_{i+1}-T_i)$ is a scaler. Specifically:

$$\mathbb{S} = \begin{bmatrix} \widehat{\sigma}^2(T_i,\widehat{K}_0) & & \\ \vdots & \ddots & \vdots \\ & & \widehat{\sigma}^2(T_i,\widehat{K}_N) \end{bmatrix} \tag{C.3}$$

$$\mathbb{D} = \begin{bmatrix} 0 & 0 & & & & & \\ l_1 & -l_1-u_1 & u_1 & & & & \\ & l_2 & -l_2-u_2 & u_2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & l_{N-1} & -l_{N-1}-u_{N-1} & u_{N-1} & \\ & & & & 0 & 0 & \end{bmatrix} \tag{C.4}$$

where

$$l_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}} \frac{1}{\widehat{K}_j - \widehat{K}_{j-1}}$$

$$u_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}} \frac{1}{\widehat{K}_{j+1} - \widehat{K}_j}$$

Given the price vector at T_i as the input:

$$\left[\widehat{C}(T_i,\widehat{K}_0),\ \ \widehat{C}(T_i,\widehat{K}_1),\ \ \widehat{C}(T_i,\widehat{K}_2),\ \ \cdots\ \ \widehat{C}(T_i,\widehat{K}_N)\right]$$

and the matrix $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.))$, we can compute the price vector at T_{i+1}:

$$\left[\widehat{C}(T_{i+1},\widehat{K}_0),\ \ \widehat{C}(T_{i+1},\widehat{K}_1),\ \ \widehat{C}(T_{i+1},\widehat{K}_2),\ \ \cdots\ \ \widehat{C}(T_{i+1},\widehat{K}_N)\right].$$

We want the price vector at T_{i+1} produced by the above LVF to match the price vector we computed using SABR model on T_{i+1}. In other words, we will try to find the $\widehat{\sigma}(T_i,.)$, which has the form as in equation (C.1), by solving the following non-linear least square problem:

$$\inf_{\widehat{\sigma}(T_i,.)} \sum_j \left(\frac{\widehat{C}(T_{i+1},\widehat{K}_j) - \widehat{C}_{SABR}(T_{i+1},\widehat{K}_j)}{Vega_B(T_{i+1},\widehat{K}_j)}\right)^2 \tag{C.5}$$

where we set:

$$\widehat{C}_{SABR}(T_{i+1},\widehat{K}_j)=\frac{\widetilde{C_{SABR}}(T_{i+1},K_j)}{D(t,T_{i+1})F(t,T_{i+1})},$$

$\widehat{Vega_B}(T_{i+1},\widehat{K}_j)$ is the vega computed using SABR model calibrated to market prices at T_{i+1} and $\widetilde{C_{SABR}}(T,K)$ is the arbitrage-free SABR model value we produce in previous section.

Note that for the intial case $T_0=t$, $\widehat{C}(T_0,\widehat{K})=max(1-\widehat{K},0)$ is given as the payoff. Given $\widehat{C}(T_0,\widehat{K})$, we firstly solve (C.5) for the $\widehat{\sigma}(T_0,.)$. After obtaining $\widehat{\sigma}(T_0,.)$, we can then solve the forward system (C.5) to get

$$\begin{bmatrix}\widehat{C}(T_i,\widehat{K}_0)\\ \widehat{C}(T_i,\widehat{K}_1)\\ \widehat{C}(T_i,\widehat{K}_2)\\ \vdots\\ \widehat{C}(T_i,\widehat{K}_N)\end{bmatrix}$$

sequentially for $i=1,2,\ldots,T_{M-1}$, where M is the number of expiries in the grid.

After we solved for $\widehat{\sigma}(T_i,.),i=0,1,\ldots,T_{M-1}$, for $T\in(T_i,T_{i+1}]$, we can fill in the gaps by:

$$\begin{bmatrix}\widehat{C}(T,\widehat{K}_0)\\ \widehat{C}(T,\widehat{K}_1)\\ \widehat{C}(T,\widehat{K}_2)\\ \vdots\\ \widehat{C}(T,\widehat{K}_{max})\end{bmatrix}=\mathbb{M}^{-1}(T,T_i,\widehat{\sigma}(T_i,.))\begin{bmatrix}\widehat{C}(T_i,\widehat{K}_0)\\ \widehat{C}(T_i,\widehat{K}_1)\\ \widehat{C}(T_i,\widehat{K}_2)\\ \vdots\\ \widehat{C}(T_i,\widehat{K}_{max})\end{bmatrix}\tag{C.6}$$

where $\mathbb{M}(T,T_i,\widehat{\sigma}(T_i,.))=I-\mathbb{SD}(T-T_i)$ is now a tri-diagonal matrix parametrized by $\sigma(T_i,.)$, T and T_i. Note that $\sigma(T_i,.)$ is known after the calibration. We then recover the call price by:

$$C(T,K)=\widehat{C}(T,\widehat{K})D(t,T)F(t,T)$$

The finite difference scheme (C.2) and (C.6) produce call option prices that are free of butterfly and calendar arbitrage. We prove this with the following Theorem C.1 and Theorem C.2. In this thesis, we provide proofs for discrete cases with a given grid of strikes and expiries. For detailed proofs for the continuous cases, interested readers can refer to [6].

Theorem C.1. *On the discrete strike grid $\widehat{K}_0,\ldots,\widehat{K}_N$, the finite difference scheme (C.2) and (C.6) produces call option prices that are convex in strikes:*

$$C(T,K_{i-1})-C(T,K_i)>\frac{K_i-K_{i-1}}{K_{i+1}-K_i}\left(C(T,K_i)-C(T,K_{i+1})\right)\ \textit{when}\ K_{i+1}>K_i>K_{i-1}$$

Proof. We prove this by the mathematical induction.
Base case: For the first expiry $T_0=t$, we have $\widehat{C}(T_0,\widehat{K})=max(1-\widehat{K},0)$ which is the normalized

payoff. We naturally have the convexity in strikes at T_0. Denote the vector of option value to be:

$$\widehat{\mathbf{C}}(T_0) = \left[\widehat{C}(T_0,\widehat{K}_0) \quad \widehat{C}(T_0,\widehat{K}_1) \quad \widehat{C}(T_0,\widehat{K}_2) \quad \cdots \quad \widehat{C}(T_0,\widehat{K}_N)\right]$$

As we can see, the convexity in strikes at T_0 implies all elements in $\mathbb{D}\widehat{\mathbf{C}}(T_0)$ are non-negative.
Induction step: Assume we have the convexity in strikes at $T_i, i = 0,1,\ldots,N-1$, i.e., all elements in $\mathbb{D}\widehat{\mathbf{C}}(T_i)$ are non-negative. The finite difference scheme (C.6) can be written as the matrix equation system

$$(I - \mathbb{S}\mathbb{D}(T-T_i))\widehat{\mathbf{C}}(T) = \widehat{\mathbf{C}}(T_i),\ i = 0,1,2,\ldots,N-1,\ T \in (T_i, T_{i+1}]. \tag{C.7}$$

Thus, we can rewrite the equation (C.7) as

$$[\mathbb{S}^{-1} - \widetilde{\mathbb{D}}](\mathbb{S}\widetilde{\mathbb{D}}\widehat{\mathbf{C}}(T)) = \widetilde{\mathbb{D}}\widehat{\mathbf{C}}(T_i) \tag{C.8}$$

where we denote $\widetilde{\mathbb{D}} = \mathbb{D}(T-T_i)$.

We assume that $\widehat{\sigma}(T_i,\widehat{K}) > 0$ for all $\widehat{K}$, therefore we can easily see that:

$$\mathbb{A} = \mathbb{S}^{-1} - \widetilde{\mathbb{D}}$$

is an M-Matrix [150] and thus all elements of $\mathbb{A}^{-1}$ are non-negative.

We can then easily see that if elements in the vector $\mathbb{D}\widehat{\mathbf{C}}(T_i)$ are all non-negative, then elements in $\widetilde{\mathbb{D}}\widehat{\mathbf{C}}(T_i)$ are all non-negative. With all elements of $\mathbb{A}^{-1}$ being non-negative, we can easily show that elements in the vector $\mathbb{S}\widetilde{\mathbb{D}}\widehat{\mathbf{C}}(T)$ are all non-negative. Furthermore, since:

$$\mathbb{S}\widetilde{\mathbb{D}}\widehat{\mathbf{C}}(T) = (T-T_i)\mathbb{S}\mathbb{D}\widehat{\mathbf{C}}(T),$$

and $\mathbb{S}$ is a diagonal matrix wih non-negative diagonal elements, we can easily see that all elements in $\mathbb{D}\widehat{\mathbf{C}}(T)$ are non-negative which is equivalent to show that:

$$C(T,K_{i-1}) - C(T,K_i) > \frac{K_i - K_{i-1}}{K_{i+1}-K_i}\left(C(T,K_i) - C(T,K_{i+1})\right).$$

Conclusion: Since both the base case and the induction step have been proved as true, by mathematical induction, we therefore prove the convexity in strikes which proves the absence of butterfly arbitrage.

□

Theorem C.2. *On the discrete strike grid $\widehat{K}_0,\ldots,\widehat{K}_N$, the finite difference scheme (C.2) and (C.6) produces call option prices increasing in maturity:*

$$C(T_i,K_i) \geq C(T_{i-1},K_i) \ \textit{ when } T_i > T_{i-1}$$

Proof. Recall that, for $T \in (T_i,T_{i+1}], i = 0,\ldots,M-1$, we rewrite the finite difference scheme

(C.6) as:

$$[I - \mathbb{SD}(T - T_i)]\widehat{\mathbf{C}}(T) = \widehat{\mathbf{C}}(T_i) \tag{C.9}$$

Differentiating equation (C.9) by T on both side:

$$\frac{\partial[I - \mathbb{SD}(T - T_i)]}{\partial T}\widehat{\mathbf{C}}(T) + [I - \mathbb{SD}(T - T_i)]\frac{\partial \widehat{\mathbf{C}}(T)}{\partial T} = 0$$

Therefore:

$$[I - \mathbb{SD}(T - T_i)]\frac{\partial \widehat{\mathbf{C}}(T)}{\partial T} = \mathbb{SD}\widehat{\mathbf{C}}(T)$$

Multiply S^{-1} on both side:

$$[S^{-1} - \mathbb{D}(T - T_i)]\frac{\partial \widehat{\mathbf{C}}(T)}{\partial T} = \mathbb{D}\widehat{\mathbf{C}}(T) \tag{C.10}$$

Recall we define:

$$\mathbb{A} = \mathbb{S}^{-1} - \widetilde{\mathbb{D}}$$

where we denote $\widetilde{\mathbb{D}} = \mathbb{D}(T - T_i)$. Therefore, we can rewrite equation (C.10) as:

$$\frac{\partial \widehat{\mathbf{C}}(T)}{\partial T} = \mathbb{A}^{-1}\mathbb{D}\widehat{\mathbf{C}}(T) \tag{C.11}$$

Since in Theorem C.1 we already prove the convexity in strikes which implies elements in $\mathbb{D}\widehat{\mathbf{C}}(T)$ are all non-negative and all elements in $\mathbb{A}^{-1}$ are non-negative since $\mathbb{A}$ is an M-Matrix [150], we can further conclude that all elements in $\frac{\partial \widehat{\mathbf{C}}(T)}{\partial T}$ are non-negative. Therefore we prove the generated option prices are increasing in maturity which proves the absence of calendar arbitrage. □

Appendix D

Model Structure of $\mathbf{GRU}_{\mathbf{TOTAL}}$ and $\mathbf{GRU}^{\mathbf{LOCAL}}_{\mathbf{TOTAL}}$

Below we briefly illustrate the model structure of $\mathrm{GRU}_{\mathrm{TOTAL}}$ and $\mathrm{GRU}^{\mathrm{LOCAL}}_{\mathrm{TOTAL}}$

D.1 Feature Selection via Embedded Feature Weighting

Similarly as in GRU_{δ}, for the sequential feature $\mathbf{y}^{T,K}_{\check{\mathrm{t}}_i}$, the j^{th} component of the normalized weight vector is given by

$$\frac{exp(\omega^S_j)}{\sum_{i=1}^{d_s} exp(\omega^S_i)}$$

The weighted feature vector at time $\check{\mathrm{t}}_i$ is defined as

$$\widehat{\mathbf{y}}^{T,K}_{\check{\mathrm{t}}_i} = \frac{exp(\omega^S)}{\sum_{j=1}^{d_s} exp(\omega^S_j)} \odot \mathbf{y}^{T,K}_{\check{\mathrm{t}}_i}$$

D.2 GRU Encoder

At the step i, the encoder computes the value of the hidden state $\mathbf{h}_i$ using a GRU cell. The input at the step i of the encoder is $\widehat{\mathbf{y}}^{T,K}_{\check{\mathrm{t}}_i}$, $i = 1,\ldots,N+1$. The internal structure of the GRU cell is shown in Figure 4.2.

Let $\boldsymbol{W}_z, \boldsymbol{U}_z, \mathbf{b}_z, \boldsymbol{W}_r, \boldsymbol{U}_r, \mathbf{b}_r, \boldsymbol{W}_h, \boldsymbol{U}_h, \mathbf{b}_h$ denote parameters shared by all GRU cells. We com-

pute the $\mathbf{h}_i$ as:

$$\begin{aligned}
\mathbf{z}_i &= sigmoid(\boldsymbol{W}_z \widehat{\mathbf{y}}^{T,K}_{\check{\mathrm{t}}_i} + \boldsymbol{U}_z \mathbf{h}_{i-1} + \mathbf{b}_z) \\
\mathbf{r}_i &= sigmoid(\boldsymbol{W}_r \widehat{\mathbf{y}}^{T,K}_{\check{\mathrm{t}}_i} + \boldsymbol{U}_r \mathbf{h}_{i-1} + \mathbf{b}_r) \\
\widehat{\mathbf{h}}_i &= tanh(\boldsymbol{W}_h \widehat{\mathbf{y}}^{T,K}_{\check{\mathrm{t}}_i} + \boldsymbol{U}_h (\mathbf{r}_i \odot \mathbf{h}_{i-1}) + \mathbf{b}_h) \\
\mathbf{h}_i &= (1 - \mathbf{z}_i) \odot \mathbf{h}_{i-1} + \mathbf{z}_i \odot \widehat{\mathbf{h}}_i
\end{aligned}$$

The hidden state at the last step $\mathbf{h}_{N+1}$, corresponding to time $\check{\mathrm{t}}_{N+1} = t$, is supplied to the decoder as the fixed size vector $\widehat{\mathbf{h}}_E$, which extracts relevant information in $\mathbf{Y}^{T,K}_t$.

D.3 Decoder for $\mathbf{GRU}_{\mathbf{TOTAL}}$

The decoder of $\mathrm{GRU}_{\mathrm{TOTAL}}$ computes the candidate output $\widehat{\delta}^M_{t,T,K}$ in the following way:

$$\widehat{\delta}^M_{t,T,K} = sigmoid(\mathbf{v}^T_{out}\ tanh(\boldsymbol{U}_{out} \widehat{\mathbf{h}}_E + \boldsymbol{W}_{out} \delta^M_{t-\Delta t,T,K} + \mathbf{b}_{out})).$$

The the output gate value W_δ is given by:

$$W_\delta = sigmoid(\mathbf{v}^T_{Gate}\ tanh(\boldsymbol{U}_{Gate} \widehat{\mathbf{h}}_E + \boldsymbol{W}_{Gate} \delta^M_{t-\Delta t,T,K} + \mathbf{b}_{Gate})).$$

For hedging a call option, the final output from GRU_δ is :

$$\delta^M_{t,T,K} = \widehat{\delta}^M_{t,T,K} \times W_\delta + \delta^{BS}_{t,T,K} \times (1 - W_\delta)$$

For hedging a put option, the final output from the model is:

$$\delta^M_{t,T,K} = -\widehat{\delta}^M_{t,T,K} \times W_\delta + \delta^{BS}_{t,T,K} \times (1 - W_\delta)$$

where $\widehat{\delta}^M_{t,T,K}$ is the candidate output.

Appendix E

Comparison of $\mathbf{GRU}_{\mathbf{TOTAL}}$ and $\mathbf{GRU}^{\mathbf{LOCAL}}_{\mathbf{TOTAL}}$

We define the total hedging risk as:

$$\text{Risk}^{total}_{t_0,T,K} = \sum_{j=0}^{N_{rb}-1} \left\{ \left[\frac{S_{t_{j+1}}}{D(t_{j+1},T)} - \frac{S_{t_j}}{D(t_j,T)} \right] \delta^M_{t_j,T,K} \right\} + \frac{V_{t_0,T,K}}{D(t_0,T)} - V_{T,T,K} \tag{E.1}$$

where $D(t,T) = e^{-r(T-t)}$ is the discount factor and $\{t_0, t_1, \ldots, t_{N_{rb}-1}\}$ is the set of rebalacing time.

In training the $\text{GRU}_{\text{TOTAL}}$, we use the following objective:

$$Obj_{total} = \sum_{i=1}^{M} \left| \text{Rel}^{total}_{t_0^i,T^i,K^i} \right| \tag{E.2}$$

where the relative total hedging error is defined as:

$$\text{Rel}^{total}_{t_0,T,K} = \frac{D(t_0,T_i)\text{Risk}^{total}_{t_0,T,K}}{V_{t_0,T,K}} \tag{E.3}$$

The objective for training the $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is:

$$Obj_{Local} = \sum_{i=1}^{M} \sum_{t \in \mathbf{t}^i_{RB}} |\Delta V^{mkt}_{t,T^i,K^i} - \Delta S_t \delta^M_{t,T^i,K^i}| \tag{E.4}$$

where $\mathbf{t}^i_{RB} = \{t^i_0, \ldots, t^i_{N_{rb}-1}\}$ is the set of rebalancing dates for the i-th hedging scenarios with

expiry T^i and initial date t_0^i and we have:

$$\begin{aligned}\Delta V_{t_j,K,T} &= D(t_0,t_{j+1})V_{t_{j+1},K,T} - D(t_0,t_j)V_{t_j,K,T}\\ \Delta S_{t_j} &= D(t_0,t_{j+1})S_{t_{j+1}} - D(t_0,t_j)S_{t_j}\\ D(t,T) &= e^{-r(T-t)}\end{aligned}$$

$$t_j = t_0 + j\Delta t;\ j = 0,\ldots,N_{rb}-1;\ t_0 = T - N_{rb}\Delta t$$

The model structure for $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is the same which is discussed in Appendix D. The same set of hedging scenarios are used as the training, testing and validation data sets. The training procedure is also the same as indicated in Algorithm 8. The only difference is the objective function used in training.

E.1 Call Option Total Hedging Comparison

In this subsection, we present the results for call options. We show the hedging performance for Near-The-Money(NTM), In-The-Money(ITM), Out-of-The-Money(OTM) separately. Note that we are not training models for NTM, ITM and OTM separately. We still train the model using all training set. The NTM, OTM, and ITM scenarios are classified based on the Black-Scholes delta at the initial date t_0 where we set up the hedging portfolio: $\delta^{BS}_{t_0,T,K}$. For call option, the criteria is:

- NTM: $0.3 \leq \delta^{BS}_{t_0,T,K} < 0.7$
- ITM: $0.7 \leq \delta^{BS}_{t_0,T,K} < 0.95$
- OTM: $0.05 \leq \delta^{BS}_{t_0,T,K} < 0.3$

We omit the testing scenarios for deep in-the-money and deep out-of-the money options due to the fact that they are highly illiquid in market and their market quotes are highly unreliable. Also, the deep in-the-money and deep out-of-the money scenarios are deleted from training set and validation set.

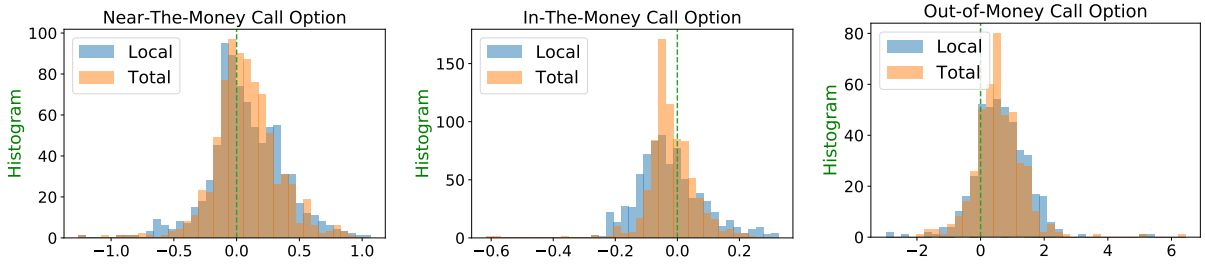
- Deep ITM: $0.95 \leq \delta^{BS}_{t_0,T,K} < 1.0$
- Deep OTM: $0.0 \leq \delta^{BS}_{t_0,T,K} < 0.05$

E.1.1 Call Option Weekly Hedging Comparison

In Table E.1, we demonstrate the results on weekly hedging call options. Furthermore, in Figure E.1, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the relative hedging error of $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$.

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	**0.1927**	**0.0571**	**0.7344**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.2250	0.0866	0.8285
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.2827**	**0.1121**	**0.5298**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.3622	0.1806	0.5753
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.4721**	**0.1865**	**1.0003**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.5643	0.2081	1.1673
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**0.5301**	**0.1976**	1.5077
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.6361	0.2168	**1.3583**
CVaR (99%)	$\text{GRU}_{\text{TOTAL}}$	0.8205	0.3261	**1.6090**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	**0.7942**	**0.2301**	2.1206

Table E.1: Summary of weekly hedging S&P 500 call options (testing set) for 100 business days with total hedging evaluation criteria described in section 6.2. Please note that the total hedging evaluation in this table assumes we are at the sell-side of the option trading.



(a) (b) (c)

Figure E.1: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and local risk hedging model $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ on weekly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

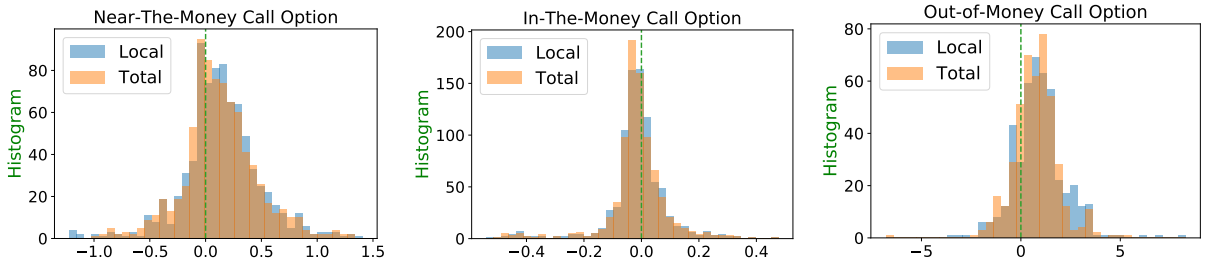
From Table E.1, we can see that, $\text{GRU}_{\text{TOTAL}}$ performs better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of most of total risk measures except for the CVaR(99%) for ITM and NTM scenarios and VaR(99%) for the OTM scenarios. We have observed significant reduction of the mean absolute relative error $\text{GRU}_{\text{TOTAL}}$ when comparing with $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. However, in terms of tail loss reduction, the improvement from $\text{GRU}_{\text{TOTAL}}$ over $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is less significant.

E.1.2 Call Option Monthly Hedging Comparison

In Table E.2, we demonstrate the results on monthly hedging call options. Furthermore, in Figure E.2, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the relative hedging error of $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. From Table E.2, we can see that, $\text{GRU}_{\text{TOTAL}}$ still

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	**0.2643**	**0.0633**	**1.0479**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.2740	0.0642	1.2255
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	0.4102	0.1472	1.0842
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	**0.3998**	**0.1394**	**0.9531**
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.6073**	**0.3125**	**1.6658**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.6671	0.3144	1.6962
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**0.7752**	**0.4300**	**1.7567**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.8703	0.4329	1.9142
CVaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**0.8692**	**0.4627**	2.7536
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	1.0861	0.4670	**2.5194**

Table E.2: Summary of monthly hedging S&P 500 call options (testing set) for 100 business days with total risk hedging evaluation criteria described in section 6.2. The total hedging evaluation in this table assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure E.2: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and local risk hedging model $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ on monthly hedging S&P 500 call options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

performs better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of reducing the mean absolute relative error. However, the reduction is less significant than the comparison in weekly hedging. Also, $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ perform roughly the same in terms of tail loss measured by VaR and CVaR.

E.2 Put Option Total Risk Hedging Comparison

In this subsection, we present the results for put options. We again show the hedging performance for Near-The-Money(NTM), In-The-Money(ITM), Out-of-The-Money(OTM) separately. The NTM, OTM, and ITM scanerios are classified based on the Black-Scholes delta at the initial date t_0 where we set up the hedging portfolio: $\delta^{BS}_{t_0,T,K}$. For put option, the criteria is:

- NTM: $-0.3 \geq \delta^{BS}_{t_0,T,K} > -0.7$
- ITM: $-0.7 \geq \delta^{BS}_{t_0,T,K} > -0.95$
- OTM: $-0.05 \geq \delta^{BS}_{t_0,T,K} > -0.3$

We omit the testing scenarios for deep in-the-money and deep out-of-the money options due to the fact that they are highly illiquid in market and their market quotes are highly unreliable. Also, the deep in-the-money and deep out-of-the money scenarios are deleted from training set and validation set.

- Deep OTM: $0.0 \geq \delta^{BS}_{t_0,T,K} > -0.05$
- Deep ITM: $-0.95 \geq \delta^{BS}_{t_0,T,K} > -1.0$

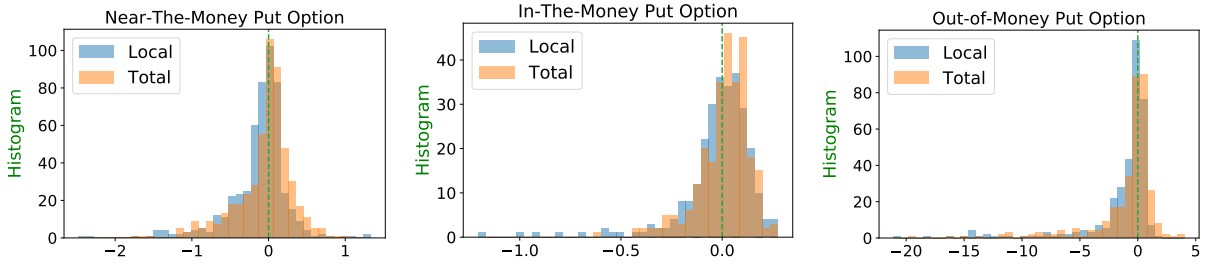
E.2.1 Put Option Weekly Hedging Comparison

In Table E.3, we demonstrate the results on monthly hedging put options. Furthermore, in Figure E.3, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the relative hedging error of $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$.

From Table E.3, we can see that, $\text{GRU}_{\text{TOTAL}}$ still performs better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ in terms of reducing the mean absolute relative error for ITM and OTM scenarios and the performance for NTM scenarios is similar. On the other hand, in terms of tail loss measured by VaR and CVaR, the reduction from $\text{GRU}_{\text{TOTAL}}$ over $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ is significant for ITM scenarios.

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	0.2535	**0.0965**	**1.5356**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	**0.2516**	0.1140	1.6042
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.8124**	**0.2364**	**7.2478**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.8229	0.3160	8.0506
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**1.0475**	**0.3452**	**10.9438**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	1.2335	0.5405	11.8778
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**1.1138**	**0.3763**	**11.7573**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	1.4361	0.7996	14.5369
CVaR(99%)	$\text{GRU}_{\text{TOTAL}}$	**1.3597**	**0.4616**	**15.1555**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	1.7732	0.9739	17.0642

Table E.3: Summary of weekly hedging S&P 500 put options (testing set) for 100 Business days with total risk hedging evaluation criteria described in section 6.2. Please note that the total hedging evaluation in this table assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure E.3: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and local risk hedging model $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ on weekly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

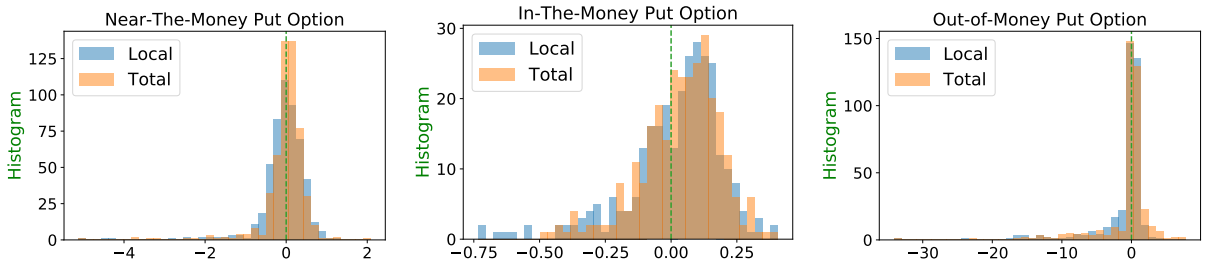
E.2.2 Put Option Monthly Hedging Comparison

In Table E.4, we demonstrate the results on monthly hedging put options. Furthermore, in Figure E.4, we compare the distribution of the relative hedging error of $\text{GRU}_{\text{TOTAL}}$ with the distributions of the relative hedging error of $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$.

From Table E.4, we can see that, for NTM and ITM scenarios, we achieve better mean absolute relative error from $\text{GRU}_{\text{TOTAL}}$. For OTM scenarios, $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ performs better in terms of mean absolute relative error. The tail loss measured by VaR and CVaR for NTM scenarios is roughly the same for $\text{GRU}_{\text{TOTAL}}$ and $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. $\text{GRU}_{\text{TOTAL}}$. The tail loss from $\text{GRU}_{\text{TOTAL}}$ for OTM scenarios is slightly better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$. The tail loss from $\text{GRU}_{\text{TOTAL}}$ for ITM scenarios is significantly better than $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$.

		Near-The-Money	In-The-Money	Out-of-The-Money
Mean Abs Relative Error	$\text{GRU}_{\text{TOTAL}}$	**0.2986**	**0.1240**	1.7639
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.3485	0.1394	**1.6849**
VaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**0.7395**	**0.2562**	8.5602
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	0.7558	0.3268	**8.1812**
CVaR (95%)	$\text{GRU}_{\text{TOTAL}}$	**1.7761**	**0.3577**	**13.3160**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	1.8144	0.4898	14.6857
VaR (99%)	$\text{GRU}_{\text{TOTAL}}$	2.1792	**0.4121**	**15.2323**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	**2.1577**	0.6454	16.5192
CVaR (99%)	$\text{GRU}_{\text{TOTAL}}$	**3.4001**	**0.4509**	**20.6503**
	$\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$	3.3717	0.6910	21.7928

Table E.4: Summary of monthly hedging S&P 500 put options for 100 business days with total hedging evaluation criteria described in section 6.2. The total hedging evaluation in this table assumes we are on the sell-side of the option trading.



(a) (b) (c)

Figure E.4: Comparing total risk hedging model $\text{GRU}_{\text{TOTAL}}$ and local risk hedging model $\text{GRU}^{\text{LOCAL}}_{\text{TOTAL}}$ on monthly hedging put options (testing set) in terms of the distribution of the relative hedging portfolio value at the expiries as in equation (6.1.3). The distribution in this figure assumes we are on the sell-side of the option trading.

Appendix F

No-Arbitrage Price Surface

In this section, we describe how to calibrate an arbitrage-free price surface on each business day from the market option prices. We want to create a parametrization of the price surface that is **arbitrage-free**. We use use SABR model to match the market volatility smiles and we use an arbitrage-free interpolation based on Local Volatility Function (LVF) model to interpolate the SABR model value between expiries.

In this thesis, we choose the SABR model over LVF models in matching market smiles and computing the value of options with the strikes unobserved in the market. This is because LVF models that were initially proposed by Dupire et al. [68], Derman and Kani [60] and Rubinstein [158] and put into highly efficient pricing engines by Andersen and Brotherton-Ratcliffe [5] and Dempster and Richards [59] amongst others, heavily rely on an arbitrage-free BS implied volatility surface as the input. If there are arbitrage violations, the convergence of the algorithm solving the underlying generalized Black Scholes partial differential equation will be obstructed [75]. Unfortunately, an arbitrage-free BS implied volatility surface as the input is not guranteed in practice. For instance, we in this thesis observe market bid and ask quotes of European options. The mid prices are used in calibrating pricing functions. The input of BS implied volatilities from mid prices are not guaranteed to be arbitrage-free. Therefore, we need to remove the arbitrage of the BS implied volatilities in the input data either manually or use arbitrage-free smoonthing algorithm as in [75] before using it as the input in LVF pricing. Another problem as indicated by Hagan et al. [99] is that the dynamics of the market smile predicted by LVF is opposite to the market behavior. The contradiction between model and market lead to unstable sensitivities (delta, vega) computation. Hedging using the delta from LVF may perform worse than hedging using BS implied delta [99]. [1]

Surface calibration models based on LVF that do not reply on the assumption that the input BS implied volatility surface is arbitrage-free exists. For example, volatility interpolation algorithm from Andreasen and Huge [6] can create an arbitrage-free surface without assuming the input is arbitrage-free. However, the resulting model cannot interpolate and extrapolate in strikes

[1] Note that we also use SABR-Bartlett delta as the comparing hedging method, we will need to calibrate the SABR model for each expiry anyway.

as we will discuss in more details in later section. Therefore, in this thesis, we only use the volatility interpolation algorithm [6] to interpolate the model value from SABR between market available expiries.

F.0.0.1 Arbitrage-Free Surface From SABR Model

In this section, we discuss how to use SABR model to create an arbitrage-free surface calibrated to match market available prices. Although SABR model is computationally efficient and can match the market volatility smile well, it is not arbitrage-free. The formula (2.2.11) is an approximation, obtained from an asymptotic series expansion. Its accuracy degrades if the option strikes move away from the option at-the-money (ATM) strike. Therefore, we also discuss how to fix the arbitrage issue of SABR.

Following [85], we check whether an implied volatility surface is free of *calendar arbitrage*, and *butterfly arbitrage*, which we describe below. Assume that we have a collection of European call option prices $\{C(T,K)\}_{T,K}$ for a range of strikes, K, and expiries, T, with the Black-Scholes implied volatility surface $\{\sigma^{imp}(T,K)\}_{T,K}$. We also suppose that interest rates are deterministic with $D(t,T) = e^{-r(T-t)}$ for the discount factor, where t is trading date and T is the maturity date.

1. Butterfly Arbitrage: At time t, given a collection of call option prices $\{C(T,K)\}_{T,K}$, using Dupire's method [68], one can write the option value with an implied probability density $p(\cdot;T,S_t)$ such that

$$C(T,K) = D(t,T)\int_{(0,\infty)}(S-K)_+ p(S;T,S_t)dS.$$

   We say that the surface $\{\sigma^{imp}(T,K)\}_{T,K}$, where $\sigma^{imp}(T,K)$ is the Black-Scholes implied volatility, is free of Butterfly Arbitrage if its implied probability density $p(\cdot;T,S_t)$ is a valid density, i.e., $p(S;T,S_t) \geq 0$ for all $S > 0$ and $\int_0^\infty p(S;T,S_t)dS = 1$. Additionally, the condition $p(S;T,S_t) \geq 0$ for all $S > 0$ is equivalent to require $\frac{\partial^2 C(T,K)}{\partial K^2} \geq 0$ for all $K > 0$ since Breeden and Litzenberger [24] show that:

$$\left.\frac{\partial^2 C(T,K)}{\partial K^2}\right|_{K=x} = D(t,T)p(x;T,S_t)$$

2. Calendar Arbitrage: Given a surface $\sigma^{imp}(T,K)$, we consider, at the time t, the corresponding total variance (TV) surface defined by

$$w(\tau,k) = \sigma^{imp}(T,K)^2\tau$$

   where $\tau = T-t$ and k is parameterized by log-moneyness, i.e. $k := \log(K/F(t;T))$, and $F(t;T) = S_t e^{(r-q)(T-t)}$ is the at-the-money forward (ATMF) price for S_t. Let $t = T_0 < T_1 < \cdots < T_M$ be a set of expiries. We say that the surface $\{\sigma^{imp}(T,K)\}_{T,K}$ is free of Calendar Arbitrage

- if $\frac{\partial w(\tau,k)}{\partial T} \geq 0$ for all $k \in \mathbb{R}, T > 0$ for continuous time data
- if $w(\tau_i,k) \leq w(\tau_{i+1},k)$ for all $k \in \mathbb{R}, T_i < T_{i+1}$ for discrete time data

Furthermore, given a grid of strikes: $0 = K_0 < K_1 < \cdots < K_N$, and a grid of expiries $t = T_0 < T_1 < \cdots < T_M$. The corresponding discrete criteria [33] for a grid of option prices to be free of arbitrage are set as the following with $j = 1, \ldots, N-1$:

1. No butterfly spread arbitrage condition:

$$C(T_j, K_{i-1}) - C(T_j, K_i) > \frac{K_i - K_{i-1}}{K_{i+1} - K_i} \left(C(T_j, K_i) - C(T_j, K_{i+1}) \right) \tag{F.1}$$

2. No calendar spread arbitrage:

$$C(T_j, K_i) \geq C(T_{j-1}, K_i) \tag{F.2}$$

In this thesis, we will use SABR model to obtain the option price for an strike K unobserved in the market when T is an expiry listed in the exchange.

We denote the grid of market observed expiries on a business date t to be

$$\mathbf{T}_t^{mkt} = \{T_0 < T_1 < \cdots < T_M\}$$

We set the first expiry be t: $T_0 = t$. Note that for the first expiry $T_0 = t$, the market option value at t is just the option payoff at t which is $\max\{S_t - K, 0\}$ (call) or $\max\{K - S_t, 0\}$ (put). For an expiry T_i, let us further define:

$$\mathbf{K}^{mkt}(t,T) = \{K_{t,T,1}^{mkt}, \ldots, K_{t,T,N_K}^{mkt}\}$$

to be the grid of strikes K observed in market on date t for which we have market option value for the expiry T

To calibrate an option value function with the market prices under SABR model, given T_i and fix $\beta = 1$, we solve

$$\min_{\alpha,\nu,\rho} \sum_{K \in \mathbf{K}^{mkt}(t,T_i)} \left(V_{SABR}(S_t, t, T_i, K, r, q; \alpha, \beta, \nu, \rho) - V_{t,T_i,K}^{mkt} \right)^2$$

where $V_{SABR}(S,t,T,K,r,q;\alpha,\beta,\nu,\rho)$ is option pricing function described in section 2.2.3 with the approximation formula (2.2.11). Hagan et al. [99] suggest that β can be chosen from prior beliefs about which assumption on the distribution of S_T is appropriate (e.g., $\beta = 0$ implies a normal distribution of S_T conditioned on a realization of the volatility while $\beta = 1$ implies a lognormal distribution of S_T conditioned on a realization of the volatility). In this thesis, we fix $\beta = 1$ in the calibration process since we are dealing equity options. If we are dealing with

interest rate derivatives, setting $\beta = 0$ or $\beta = 0.5$ may be more appropriate. In practice, the choice of β has little effect on the resulting shape of the volatility curve produced by the SABR model. Hagan et al. [99] suggest that the choice of β is not crucial in matching the market volatility smile. Furthermore, Bartlett [16] suggests that the choice of β is also not crucial when we use the bartlett delta as in equation (2.3.18) as the delta position from SABR model.

We choose to calibrate a different pricing function model for each expiry. A different set of parameters is specified for each expiry, describing an instantaneous process. We choose this approach because the single implied volatility surface calibrated for all expiries and strikes is unlikely to fit the actual surface very well. In addition, calibrating a single surface is harder and more time-consuming.

Note that we have three parameters to be calibrated so we need to observe at least 3 data points from market to successfully build a SABR model. Therefore, the number of strikes N_K is expected be larger than or equal to 3. [2]

Due to the fact that equation (2.2.11) being an appproximation, the implied probability density function:

$$\frac{1}{D(t,T)} \frac{\partial^2 C_{SABR}(T,K)}{\partial K^2}\bigg|_{K=x} = p(x;T,S_t) \tag{F.3}$$

where $C_{SABR}(T,K)$ is the SABR pricing function of a call option at strike K and expiry T computed using the equation (2.2.11), may become negative at very low or very high strikes. Therefore, we may observe butterfly spread arbitrage in SABR prices returned by the calibrated models. In addition, for each expiry, a separate set of SABR parameters is calibrated so that calender spread arbitrage can also exist. However, the existence of calendar arbitrage will be rare since the market option prices rarely contains calendar arbitrage and therefore the SABR model, which usually matches the market option data very well, rarely produces calendar arbitrage.

Given a grid of strikes K for which we aim to use SABR model to produce the option prices, if we found that the grid of prices returned by the SABR model has failed the **discrete** arbitrage conditions for butterfly arbitrage (F.1), we will introduce some adjustments. To fix the butterfly spread arbitrage, we implements a risk-neutral adjustment. This adjustment substitutes the two implied distribution tails by those of certain log-normal distributions. The following adjustment is inspired by [27]. Interested reader can refer to [27] for more details. Here we just briefly discuss the process of the adjustment.

Firstly, we introduce lower and upper strike limits, K_L and K_U within which the implicit probability density function (p.d.f) $p(x;T,S_t)$ from SABR model is assumed to be valid. The lower and upper strike limit can be the maximum and minimum strike K for which the discrete no butterfly spread arbitrage condition (F.1) holds. Brunner and Hafner [27] set the tail distributions

[2] The $V^{mkt}_{t,T,K}$ is the mid-price of market observed best bid and best ask prices.

as the mixture of two lognormal distributions:

$$\hat{g}(x) = \begin{cases} \lambda_L\, q(x;\mu_L^1,\sigma_L^1) + (1-\lambda_L)\, q(x;\mu_L^2,\sigma_L^2), & \text{if } 0 < x < K_L \\ p(x;T,S_t), & \text{if } K_L \le x \le K_U \\ \lambda_U\, q(x;\mu_U^1,\sigma_U^1) + (1-\lambda_U)\, q(x;\mu_U^2,\sigma_U^2), & \text{if } x > K_U \end{cases} \tag{F.4}$$

where $q(x;\mu,\sigma)$ is the p.d.f of a log-normal distribution:

$$q(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}.$$

and $p(x;T,S_t)$ is the implied probability density from the calibrated SABR model. Here, the parameters to be determined are:

$$\{\lambda_L, \mu_L^1, \sigma_L^1, \mu_L^2, \sigma_L^2, \lambda_U, \mu_U^1, \sigma_U^1, \mu_U^2, \sigma_U^2\}.$$

In this thesis, we assume the adjusted p.d.f is of the following simplified form:

$$\hat{g}(x) = \begin{cases} \lambda_L\, q(x;\mu_L,\sigma_L), & \text{if } 0 < x < K_L \\ p(x;T,S_t), & \text{if } K_L \le x \le K_U \\ \lambda_U\, q(x;\mu_U,\sigma_U), & \text{if } x > K_U \end{cases} \tag{F.5}$$

We assume that the underlying price at expiry at T: S_T is distributed according the adjusted p.d.f $\hat{g}(x)$. We choose the equation (F.5) becasue it has a simpler solution than the equation (F.4), for which we need to solve an overdetermined non-linear system.

We require the following condition to be satisfied:

1. Intergrability constraint

$$\int_0^{K_L} \lambda_L q(x;\mu_L,\sigma_L)dx + \int_{K_L}^{K_U} p(x;T,S_t)dx + \int_{K_U}^{\infty} \lambda_U\, q(x;\mu_U,\sigma_U)dx = 1 \tag{F.6}$$

2. Martingale constraint

$$\int_0^{K_L} x\lambda_L q(x;\mu_L,\sigma_L)dx + \int_{K_L}^{K_U} xp(x;T,S_t)dx + \int_{K_U}^{\infty} x\lambda_U\, q(x;\mu_U,\sigma_U)dx = F(t,T) \tag{F.7}$$

   The intergrability constraint ensures that $\hat{g}(x)$ is a valid p.d.f. The martingale constraint ensures that $E[S_T] = F(t,T) = S_t e^{(r-q)(T-t)}$ under ther adjusted p.d.f.

Since they are six unknown parameters $\{\mu_L, \mu_U, \sigma_L, \sigma_U, \lambda_L, \lambda_U\}$, additional calibration conditions are imposed. Observing that the Black-Scholes [20] model implies that, under the risk neutral measurement, the prices of the underlying asset S_T at the maturity T are log-normal distribued:

$$\ln(S_T) \sim \mathcal{N}(\ln(S_t) + (r - q - \frac{\sigma^2}{2})(T-t), \sigma^2(T-t))$$

we set $\mu_L = \ln(S_t) + (r - q - \frac{\sigma_L^2}{2})(T-t)$ and $\mu_U = \ln(S_t) + (r - q - \frac{\sigma_U^2}{2})(T-t)$ such that we have only four parameters to be solved: $\{\sigma_L, \sigma_U, \lambda_L, \lambda_U\}$. Furthermore, with $\mu_L = \ln(S_t) + (r - q - \frac{\sigma_L^2}{2})(T-t)$ and $\mu_U = \ln(S_t) + (r - q - \frac{\sigma_U^2}{2})(T-t)$, one can easily verify that if we have $\{\sigma_L = \sigma_B(K_L), \sigma_U = \sigma_B(K_U), \lambda_L = 1, \lambda_U = 1\}$, intergrability constraint (F.6) and martingale constraint (F.7) will be satisified.

We use the adjusted p.d.f (F.5) for option pricing

$$C_{\hat{g}(x)}(T,K) = D(t,T)\int_{-\infty}^{\infty}(x-K)_+\hat{g}(x)dx \tag{F.8}$$

For simplicity, we write $\sigma_B(K)$ to denote the implied Black's volatility given by the SABR formula (2.2.11) for a strike K since the other inputs in $\sigma_B(F,t,T,K;\alpha,\beta,\nu,\rho)$ as in formula (2.2.11) remain unchanged in the following discussion for an expiry T at a time t.

By setting:

$$\begin{aligned}
&\sigma_L = \sigma_B(K_L)\ ,\ \sigma_U = \sigma_B(K_U)\\
&\lambda_L = 1\ ,\ \lambda_U = 1\\
&\mu_L = \ln(S_t) + (r - q - \frac{\sigma_B(K_L)^2}{2})(T-t)\\
&\mu_U = \ln(S_t) + (r - q - \frac{\sigma_B(K_U)^2}{2})(T-t)
\end{aligned} \tag{F.9}$$

we can easily verify that (F.8) can be written as:

$$\overline{C}_{SABR}(T,K) \leftarrow C_{\hat{g}(x)}(T,K) = \begin{cases} C_{BS}(T,K;\sigma_B(K_L)) & \text{if } 0 < K < K_L \\ C_{SABR}(T,K) & \text{if } K_L \le K \le K_U \\ C_{BS}(T,K;\sigma_B(K_U)) & \text{if } K > K_U \end{cases}$$

Here we use the $\overline{C}_{SABR}(T,K)$ to indicate it is the SABR model value after the fix for the butterfly arbitrage. Since the adjusted p.d.f (F.5) with the setting (F.9) satisfies intergrability constraint (F.6) and martingale constraint (F.7) and will not be negative at tails of the distribution because that the two tails are from two log-normal distribuions, we can conclude that the adjusted $\overline{C}_{SABR}(T,K)$ will be free of butterfly arbitrage. In appendix B, we show in details that the adjusted p.d.f (F.5) with the parameters setting as in equations (F.9) will satisfy the intergrability constraint (F.6) and martingale constraint (F.7).

For calendar arbitrage, we will shift the price by the the following procedure to remove it. Suppose we have a grid of $\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$ and a grid of expiries $t = T_0 < T_1 < \cdots < T_M$. Then for each expiry $T_j, i = 1, \ldots, M$, we have:

$$shift_{T_j} = -min\left\{min_{K\in\mathbf{K}_{grid}}\left[\overline{C}_{SABR}(T_j,K) - \overline{C}_{SABR}(T_{j-1},K)\right], 0\right\}.$$

$$\widetilde{C_{SABR}}(T_j,K) \leftarrow \overline{C}_{SABR}(T_j,K) + shift_{T_j}$$

Note the shift will be zero if no calendar arbitrage is observed between T_j and T_{j-1}. Note that the constant shift will preserve the no butterfly arbitrage condition from $\overline{C}_{SABR}(T,K)$, one can easily see this by noting that no butterfly arbitrage implies that:

$$\overline{C}_{SABR}(T_j,K_{i-1}) - \overline{C}_{SABR}(T_j,K_i) > \frac{K_i - K_{i-1}}{K_{i+1} - K_i}\left(\overline{C}_{SABR}(T_j,K_i) - \overline{C}_{SABR}(T_j,K_{i+1})\right)$$

Therefore, we still have

$$\begin{aligned}&[\overline{C}_{SABR}(T_j,K_{i-1}) + shift_{T_j}] - [\overline{C}_{SABR}(T_j,K_i) + shift_{T_j}] \\ &> \frac{K_i - K_{i-1}}{K_{i+1} - K_i}\left\{\left[\overline{C}_{SABR}(T_j,K_i) + shift_{T_j}\right] - \left[\overline{C}_{SABR}(T_j,K_{i+1}) + shift_{T_j}\right]\right\}\end{aligned}$$

Lastly, there is a no call-spread arbitrage condition that requires the call option value decreasing in strikes:

$$C(T,K_{i+1}) \leq C(T,K_i) \text{ when } K_{i+1} > K_i.$$

We comments that call-spread arbitrage violation is rare and no violations of this condition from the prices produced by SABR model are observed for all the computational results in this thesis. Additionally, by the call-put parity, if we have no butterfly arbitrage and no calendar arbitrage with call option prices, then we will have no butterfly arbitrage and no calendar arbitrage in the put option prices as well.

F.0.0.2 Alternative Methods to SABR Calibration

Given BS implied volatility surface computed from market option values as a discrete set of points $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$, a natural question is whether arbitrage exists. In most cases, one must

1. Fit a parameterization to the points $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$, typically by time-slice as what we do with SABR model, obtaining a parameterization $\widetilde{\sigma}^{imp}_{model}(T_i,K)$ for each T_i separately. For example, if we fit a SABR model, $\widetilde{\sigma}^{imp}_{model}(T_i,K) = \sigma_B(F,t,T_i,K;\alpha,\beta,\nu,\rho)$ and since except K all other parameters are fixed for an expiry T_j on a time t, it is a function of K only.

2. Compute $\{\widetilde{w}(\tau_i,k)\}_{i=1}^N$ on a fine grid of k using the expression $\widetilde{w}(\tau_i,k) := \widetilde{\sigma}^{imp}_{model}(T_i,K)^2\tau_i$, where $\tau_i = T_i - t$ and k is parameterized by log-moneyness, i.e. $k := \log(K/F(t;T))$.

3. Check for the discrete no-arbitrage criteria (F.1) and (F.2) as in [33] for a given grid of strikes and a given grid of expiries.

To obtain such mappings, one typically fits a stochastic volatility model like SABR or Heston to slices of $\{\sigma^{imp}_{BS,mkt}(T,K)\}_{T,K}$ or considers a generalized model such as Stochastic Volatility Inspired (SVI) parameterizations.

The Stochastic Inspired Volatility model (SVI) [83] was used internally at Merrill Lynch and publicly disclosed by Jim Gatheral in 2004. SVI is a simple five-parameters model. In 2012, the Surface SVI (SSVI) [85] is proposed by Gatheral and Jacquier to extend SVI model to be a model that can fit the whole surface instead of just one volatility smile. The SSVI is parameterized in a way that a SSVI slice at a given maturity T is a SVI slice with only 3 parameters. This restriction leads to explicit sufficient conditions for the absence of arbitrage, while allowing enough flexibility for calibration. The SSVI model is recently extended in [102] and [53]. If the input market data used for calibration contains arbitrage, the calibrated surface from SSVI [85] or its extension [102, 53] can typically be viewed as the surface that is as close as possible to the original market data, while staying arbitrage-free. In this section we briefly review a recent variant dd-eSSVI (data-driven extended SSVI) method [53] as an example.

F.0.0.3 dd-eSSVI Parameterization

Following the work of [53], we introduce the following SSVI parameterization for a surface's Total Variance (TV), $w(\tau,k)$ as

$$w(\tau,k) = \frac{\hat{\theta}_\tau}{2}\left(1+\hat{\rho}_\tau\hat{\psi}_\tau k + \sqrt{(\hat{\psi}_\tau k+\hat{\rho}_\tau)^2+(1-\hat{\rho}_\tau^2)}\right) \tag{F.10}$$

In this parameterization we have that

- $\hat{\theta}_\tau$ is the ATM Forward TV which can be extracted from market directly.

  $$\hat{\theta}_\tau = w(\tau,0) = \sigma^{imp}_{BS,mkt}(T,K_{ATMF})^2\cdot\tau$$

  where $K_{ATMF} = F(t,T)$
- $\hat{\rho}_\tau$ controls the slope of the skew
- $\hat{\psi}_\tau$ controls the curvature, which is usually defined as a function of $\hat{\theta}_\tau$: $\hat{\psi}_\tau(\hat{\theta}_\tau)$

An important feature of this parameterizations is that it provides easy way to impose sufficient conditions on the parameters $(\hat{\theta}_\tau,\hat{\rho}_\tau,\hat{\psi}_\tau)$ so that there is no butterfly arbitrage for a given slice, and no calendar arbitrage between two time slices. Interested reader can refer to [53, 85] for the detailed conditions on those parameters.

There are many different approaches for calibrating SSVI models. For example, one can fit SSVI model to market data without imposing any constraints on the parameters and then check if any arbitrage exists by imposing the arbitrage conditions on the calibrated parameters. If arbitrage conditions are violated, one can adjust the parameters so that the sufficient conditions on parameters are satisfied. One can also use an arbitrage-free calibration [53] by imposing the sufficient conditions on the parameters into the calibration process. Interested reader can refer to [102, 53] for more details on how to calibrate the SSVI efficiently.

Since the major goal in this thesis is not to compare arbitrage-free surface calibration methods, we leave the exploration of the alternative methods to calibrate the arbitrage-free surface and comparing its impact on the data-driven risk hedging model as the future work of our study.

F.0.0.4 Volatility Interpolation Between Expiries

In the previous section, for each T_i, we calibrate a separate set of SABR parameters and we then use the calibrated SABR parameters to compute option price and the associated implied volatiliy for a predetermined grid of strikes for each T_i. We correct for butterfly arbitrage and calendar arbitrage if we detect any of them in the option values produced from the SABR models. However, after the SABR calibration, we only obtain the parametrization of option values for expiries listed in the market. Our next goal is get the parametrization of option values for expiries that are not available in the market. In this section, we discuss how to interpolate the volatility between different expiries. Note that even if we use SSVI instead of SABR model, this step of volatility interpolation between expiries is still needed since SSVI model and SABR model are both calibrated to match the volatility smile of market for each market expiry only. Under SSVI models, one usually interpolates the SSVI parameters $\{\hat{\theta}_\tau, \hat{\rho}_\tau, \hat{\psi}_\tau\}$ between different expiries available in market [53].

Andreasen and Huge [6] have introduced an efficient and arbitrage-free volatility interpolation method based on an one step finite difference implicit Euler scheme applied to a local volatility parametrization. In this thesis, we use the volatility interpolation approach to compute option price for an arbitrary expiry T unobserved in the market.

The volatility interpolation method is based on the Dupire's equation [68]. The Dupire's equation enables us to deduce the volatility function in a local volatility model from put and call options in the market. Under a risk-neutral measure, we assume:

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

where r is the risk-free interest rate and q is the dividend yield. Let $C(T,K)$ be the call option pricing function, Dupire's equation states:

$$\frac{\partial C(T,K)}{\partial T} = \frac{1}{2}\sigma^2(T,K)K^2\frac{\partial^2 C(T,K)}{\partial K^2} - (r-q)K\frac{\partial C(T,K)}{\partial K} - qC(T,K)$$

Define the normalized call price in terms of discounting factor $D(t,T)$ and forward price $F(t,T)$

and the normalized strike $\widehat{K}$ as:

$$
\begin{aligned}
D(t,T) &= e^{-r(T-t)} \\
F(t,T) &= S_t e^{(r-q)(T-t)} \\
\widehat{K} &= \frac{K}{F(t,T)} \\
\widehat{C}(T,\widehat{K}) &= \frac{C(T,\widehat{K}F(t,T))}{D(t,T)F(t,T)} = \frac{C(T,K)}{D(t,T)F(t,T)}
\end{aligned}
$$

Dupire's equation can be simplified as [6][3]:

$$
\frac{\partial \widehat{C}(T,\widehat{K})}{\partial T} = \frac{1}{2}\widehat{\sigma}^2(T,\widehat{K})\widehat{K}^2\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}, \;\; \widehat{\sigma}(T,\widehat{K}) = \sigma(T,K)
$$

Therefore, we can sequentially solve the finite difference discretization of the Dupire's forward equation using the fully implicit method. Observing that $T_0 = t$, on the trading date t, the option value expiring at t is just option payoff, we have the the initial condition $\widehat{C}(T_0,\widehat{K}) = max(1-\widehat{K},0)$. Furthermore, when $K = 0$, we arrive at the lower boundary condition $\widehat{C}(T,0) = 1$. When the largest strike $K_{max} \gg S_t$, we assume the upper boundary condition $\widehat{C}(T,\widehat{K}_{max}) = 0$ is true.

Assume we are given a grid of expiries available in market $t = T_0 < T_1 < \cdots < T_M$ and a grid of normalized strike: $0 = \widehat{K}_0 < \widehat{K}_1 < \cdots < \widehat{K}_N = \widehat{K}_{max}$, Andreasen and Huge [6] assume $\widehat{\sigma}(T,\widehat{K})$ to be a piecewise constant functions for a given T_i.

$$
\widehat{\sigma}(T_i,\widehat{K}) = \begin{cases}
\widehat{\sigma}_{T_i,\widehat{K}_0}, & \text{if } \widehat{K} \le \widehat{K}_0 \\
\vdots & \vdots \\
\widehat{\sigma}_{T_i,\widehat{K}_j}, & \text{if } \widehat{K}_{j-1} < \widehat{K} \le \widehat{K}_j \\
\vdots & \vdots \\
\widehat{\sigma}_{T_i,\widehat{K}_N}, & \text{if } \widehat{K} > \widehat{K}_N
\end{cases} \tag{F.11}
$$

The authors further assume:

$$
\left.\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}\right|_{\widehat{K}=0} = \left.\frac{\partial^2 \widehat{C}(T,\widehat{K})}{\partial \widehat{K}^2}\right|_{\widehat{K}=\widehat{K}_N} = 0 \tag{F.12}
$$

[3] We use the call option as the example but the analysis also holds for put options. More specifically, let $P(T,K)$ be the function of put option price, the Dupire's equation for put option is:

$$
\frac{\partial \widehat{P}(T,\widehat{K})}{\partial T} = \frac{1}{2}\widehat{\sigma}^2(T,\widehat{K})\widehat{K}^2\frac{\partial^2 \widehat{P}(T,\widehat{K})}{\partial \widehat{K}^2}, \;\; \widehat{\sigma}(T,\widehat{K}) = \sigma(T,K).
$$

From (F.3), one can see the second partial derivative of call prices with regards to strike K is the implied density:

$$\frac{1}{D(t,T)}\frac{\partial^2 C(T,K)}{\partial K^2}\bigg|_{K=x} = p(x;T,S_t)$$

Therefore, the boundary conditions (F.12) essentially assume that the probablity density at the low strike boundary ($K=0$) and high strike boundary ($K=K_N$) is zero, which is a reasonable assumption.

The fully implicit finite difference method in matrix form is:

$$\begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix} = (I - \mathbb{S}\mathbb{D}(T_{i+1}-T_i)) \begin{bmatrix} \widehat{C}(T_{i+1},\widehat{K}_0) \\ \widehat{C}(T_{i+1},\widehat{K}_1) \\ \widehat{C}(T_{i+1},\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_{i+1},\widehat{K}_N) \end{bmatrix} \tag{F.13}$$

where I is the identity matrix, $\mathbb{S}$ is a diagonal matrix parameterized by $\widehat{\sigma}(T_i,.)$ as in equation (F.11), $\mathbb{D}$ is proportional to the discrete second order difference matrix and $(T_{i+1}-T_i)$ is a scaler. Specifically:

$$\mathbb{S} = \begin{bmatrix} \widehat{\sigma}^2(T_i,\widehat{K}_0) & & \\ \vdots & \ddots & \vdots \\ & & \widehat{\sigma}^2(T_i,\widehat{K}_N) \end{bmatrix} \tag{F.14}$$

$$\mathbb{D} = \begin{bmatrix} 0 & 0 & & & & & \\ l_1 & -l_1-u_1 & u_1 & & & & \\ & l_2 & -l_2-u_2 & u_2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & l_{N-1} & -l_{N-1}-u_{N-1} & u_{N-1} & \\ & & & & 0 & 0 & \end{bmatrix} \tag{F.15}$$

where

$$l_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}}\frac{1}{\widehat{K}_j-\widehat{K}_{j-1}}$$

$$u_j = \frac{1}{\widehat{K}_{j+1}-\widehat{K}_{j-1}}\frac{1}{\widehat{K}_{j+1}-\widehat{K}_j}$$

Denote $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.)) = (I-\mathbb{S}\mathbb{D}(T_{i+1}-T_i))$. We can see that $\mathbb{M}(T_{i+1},T_i,\widehat{\sigma}(T_i,.))$ is a tri-diagonal matrix parametrized by $\widehat{\sigma}(T_i,.)$, T_{i+1}, and T_i.

Given the price vector at T_i as the input:

$$\begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0), & \widehat{C}(T_i,\widehat{K}_1), & \widehat{C}(T_i,\widehat{K}_2), & \cdots & \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix}$$

and the matrix $\mathbb{M}(T_{i+1}, T_i, \widehat{\sigma}(T_i, .))$, we can compute the price vector at T_{i+1}:

$$\left[\widehat{C}(T_{i+1}, \widehat{K}_0), \quad \widehat{C}(T_{i+1}, \widehat{K}_1), \quad \widehat{C}(T_{i+1}, \widehat{K}_2), \quad \cdots \quad \widehat{C}(T_{i+1}, \widehat{K}_N)\right].$$

We want the price vector at T_{i+1} produced by the above LVF to match the price vector we computed using SABR model on T_{i+1}. In other words, we will try to find the $\widehat{\sigma}(T_i, .)$, which has the form as in equation (F.11). Andreasen and Huge [6] suggest one can obtain $\widehat{\sigma}^*(T_i, .)$ by solving the following non-linear least square problem:

$$\inf_{\widehat{\sigma}(T_i,.)} \sum_{j=0}^{N} \left(\frac{\widehat{C}(T_{i+1}, \widehat{K}_j) - \widehat{C}_{SABR}(T_{i+1}, \widehat{K}_j)}{Vega_B(T_{i+1}, \widehat{K}_j)}\right)^2 \tag{F.16}$$

where we set:

$$\widehat{C}_{SABR}(T_{i+1}, \widehat{K}_j) = \frac{\widetilde{C_{SABR}}(T_{i+1}, K_j)}{D(t, T_{i+1})F(t, T_{i+1})},$$

$Vega_B(T_{i+1}, \widehat{K}_j)$ is the vega computed using SABR model calibrated to market prices at T_{i+1} and $\widetilde{C_{SABR}}(T, K)$ is the arbitrage-free SABR model value we produce in previous section.

Note that for the intial case $T_0 = t$, $\widehat{C}(T_0, \widehat{K}) = max(1 - \widehat{K}, 0)$ is given as the payoff. Given $\widehat{C}(T_0, \widehat{K})$, we firstly solve (F.16) for the $\widehat{\sigma}^*(T_0, .)$. After obtaining $\widehat{\sigma}^*(T_0, .)$, we can then solve the forward system (F.16) to get

$$\begin{bmatrix} \widehat{C}(T_i, \widehat{K}_0) \\ \widehat{C}(T_i, \widehat{K}_1) \\ \widehat{C}(T_i, \widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i, \widehat{K}_N) \end{bmatrix}$$

sequentially for $i = 1, 2, \ldots, T_{M-1}$, where M is the number of expiries in the grid.

After we solved for $\widehat{\sigma}^*(T_i, .), i = 0, 1, \ldots, T_{M-1}$, for $T \in (T_i, T_{i+1}]$, we can fill in the gaps by:

$$\begin{bmatrix} \widehat{C}(T, \widehat{K}_0) \\ \widehat{C}(T, \widehat{K}_1) \\ \widehat{C}(T, \widehat{K}_2) \\ \vdots \\ \widehat{C}(T, \widehat{K}_{max}) \end{bmatrix} = \mathbb{M}^{-1}(T, T_i, \widehat{\sigma}^*(T_i, .)) \begin{bmatrix} \widehat{C}(T_i, \widehat{K}_0) \\ \widehat{C}(T_i, \widehat{K}_1) \\ \widehat{C}(T_i, \widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i, \widehat{K}_{max}) \end{bmatrix} \tag{F.17}$$

where $\mathbb{M}(T, T_i, \widehat{\sigma}^*(T_i, .)) = I - \mathbb{SD}(T - T_i)$ is now a tri-diagonal matrix parametrized by $\sigma^*(T_i, .)$, T and T_i. Note that $\sigma^*(T_i, .)$ is known after the calibration (F.16). We then recover the call price by:

$$C(T, K) = \widehat{C}(T, \widehat{K})D(t, T)F(t, T)$$

Interpolation based on the above procedure can guarantee the option prices computed is arbitrage-

free. Detailed proofs are can be found in Appendix C.

In this thesis, we only use the above volatility interpolation algorithm to interpolate the option value produced by SABR model for expiry $T \in (T_i, T_{i+1})$, where T_i and T_{i+1} are expiries listed in the market. A natural question the reader may ask is that why we cannot apply above algorithm with purely market prices? In other words, we solve the below problem instead of problem (F.16):

$$\inf_{\widehat{\sigma}(T_i,.)} \sum_j \left(\frac{\widehat{C}(T_{i+1},\widehat{K}_j) - \widehat{C}_{mkt}(T_{i+1},\widehat{K}_j)}{Vega_B(T_{i+1},\widehat{K}_j)}\right)^2 .$$

In reality, it is hard to find a grid of normalized strike: $0 = \widehat{K}_0 < \widehat{K}_1 < \cdots < \widehat{K}_N$, for which, $\widehat{C}_{mkt}(T_i,\widehat{K}_j)$, $i = 0,1,\ldots,N$ all exists. Especially, if we want our grid of strike to cover both in-the-money option and out-of-the-money option. That is why we use SABR model which can produce option value for any K for an expiry T_i in market after the calibration. In this way, we can use any grid of strikes as we want.

Following the discussion in section F, we summarize the SABR smile calibration and the corresponding fix for the butterfly arbitrage and the calendar arbitrage in Algorithm 2 and Algorithm 3. We summarize the LVF volatility interpolation in Algorithm 4. With the help from SABR model and LVF volatility interpolation, we essentially obtain a parametrization of the option value at each trading date t: $\{V^t_{model}(T,K)\}_{T,K}$. Here the expiry T can be any value that is later than t and before the maximum expiry $T^{mkt}_{t,max}$ observed in the market on date t. The strike K price can be any value. We summarize the process of constructing the arbitrage-free options values for a given grid of strikes and a given grid of expiries in Algorithm 5.

F.0.0.5 Construction of Training, Testing and Validation Data Sets

We provide more details on training, testing and validation data sets.

Step 1: We test on the real market expiries. The set of all testing expiries is defined as:

$$\mathbf{T}_{AllTest} = \{T^{mkt} | \text{2000-01-01} \leq T^{mkt} \leq \text{2015-08-31}, T^{mkt} \text{ is a market expiry date } \}$$

Namely, we test on all the market observed expiry date T, which are between 2000-01-01 to 2015-08-31. Note that, we have included two crisis periods: the burst of dot-com bubble period (2000 to 2002) and subprime mortgage crisis period (2007 to 2008). We assume we are on the sell-side of the option trading.

Step 2: For a testing expiry date $T^{test} \in \mathbf{T}_{AllTest}$, we construct the testing set below:

$$TestSet = \{Scenario(T^{test}, K) | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T^{test})\}$$

where $\mathbf{K}^{mkt}_{grid}(t_0, T^{test})$ is the grid of market strikes for expiry T^{test} that can be observed directly from market on the initial date t_0. And t_0 is 100 business away from T^{test}. In

Algorithm 2: Function For SABR Calibration

Function `SABRCalibration(`t, T, $\mathbf{K}_{grid}$`)`**:**

Input: t: An option trading date t.
T: A **market** option expiry.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.

Extract the set of strikes available in market at time t: $\mathbf{K}^{mkt}(t,T)$

Extract the set of market option prices for expiry T at time t:$\{V^{mkt}_{t,T,K} | \forall K \in \mathbf{K}^{mkt}(t,T)\}$

Set $\beta^* = 1$

Solve:

$$\alpha^*, \nu^*, \rho^* = \text{argmin}_{\alpha,\nu,\rho} \sum_{K \in \mathbf{K}^{mkt}(t,T)} \left(V_{SABR}(S_t, t, T, K, r, q; \alpha, \beta^*, \nu, \rho) - V^{mkt}_{t,T,K} \right)^2$$

Calculate the call option prices $C_{SABR}(T,K)$ for $K \in \mathbf{K}_{grid}$ using the SABR parameters $\beta^*, \alpha^*, \nu^*, \rho^*$.

Check if there is any violation of the following condition on the grid of strikes $\mathbf{K}_{grid}$ with $i = 1, \ldots, N-1$:

$$C_{SABR}(T,K_{i-1}) - C_{SABR}(T,K_i) > \frac{K_i - K_{i-1}}{K_{i+1} - K_i} \left(C_{SABR}(T,K_i) - C_{SABR}(T,K_{i+1}) \right) \tag{F.18}$$

if *No violation* **then**
  Set $K_L = K_0$, $K_L = K_N$
else
  Set K_L to be the smallest K_i where condition (F.18) hold
  Set K_U to be the largest K_i where condition (F.18) hold
end

Calculate call option value function with no-butterfly arbitrage:

$$\overline{C}_{SABR}(T,K) \leftarrow C_{\hat{g}(x)}(T,K) = \begin{cases} C_{BS}(T,K;\sigma_B(K_L)) & \text{if } 0 < K < K_L \\ C_{SABR}(T,K) & \text{if } K_L \le K \le K_U \\ C_{BS}(T,K;\sigma_B(K_U)) & \text{if } K > K_U \end{cases}$$

`/*` $\sigma_B(K)$ `is the short form of the SABR implied volatility approximation (2.2.11).` `*/`

return $\overline{C}_{SABR}(T,K)$

Algorithm 3: Function For Returning Arbitrage Free Option Value With SABR Model

Function `SABRArbitrageFree(` t, $\mathbf{T}_{mkt}$,$\mathbf{K}_{grid}$`):`

Input: t: An option trading date t.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
`SABRCalibration`: SABR Calibration Algorithm as in Algorithm 2.

Set $\widetilde{C_{SABR}}(T_0, K) = \max(S_t - K, 0)$

`/* Option value with expiry` $T_0 = t$ `is the payoff */`

for $i = 1;\ i \leq M;\ i = i+1$ **do**

  $\overline{C}_{SABR}(T_i, K) \leftarrow$ `SABRCalibration(`t,T_i,$\mathbf{K}_{grid}$`)`

end

for $i = 1;\ i < M;\ i = i+1$ **do**

  $shift_{T_i} = -min\left\{min_{K \in \mathbf{K}_{grid}}\left[\overline{C}_{SABR}(T_i, K) - \overline{C}_{SABR}(T_{i-1}, K)\right], 0\right\}$.

  if $shift_{T_i} \neq 0$ **then**

    for $j = i;\ j < M;\ j = j+1$ **do**

      `/* We need to shift every` $T_j \geq T_i$`, if shift is not zero */`

      $\overline{C}_{SABR}(T_j, K) \leftarrow \overline{C}_{SABR}(T_j, K) + shift_{T_i}$

    end

    $\widetilde{C_{SABR}}(T_i, K) \leftarrow \overline{C}_{SABR}(T_i, K)$

  else

    $\widetilde{C_{SABR}}(T_i, K) \leftarrow \overline{C}_{SABR}(T_i, K)$

  end

end

return $\{\widetilde{C_{SABR}}(T, K) | \forall T \in \mathbf{T}_{mkt}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

Algorithm 4: Function For LVF Calibration

Function `LVFCalibration(` $t, \mathbf{T}_t^{mkt}, \mathbf{K}_{grid}, \mathbf{T}_{grid}$`):`

Input: t: An option trading date t.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
$\mathbf{T}_{grid} = \{T_0^B = t, \ldots, T_{N_B}^B\}$: A grid of expiries for outputting option value.
This grid includes every business days between $T_0 = t$ and T_M.
`SABRArbitrageFree`: SABR Surfrace Algorithm as in Algorithm 3.

$\{\widetilde{C_{SABR}}(T,K) | \forall T \in \mathbf{T}_t^{mkt}\ ,\ \forall K \in \mathbf{K}_{grid}\} \leftarrow$ `SABRArbitrageFree(`$t, \mathbf{T}_t^{mkt}, \mathbf{K}_{grid}$`)`

Calculate

$$\widehat{C}_{SABR}(T,\widehat{K}) = \frac{\widetilde{C_{SABR}}(T,K)}{D(t,T)F(t,T)}, \widehat{K} = \frac{K}{F(t,T)}$$
$$D(t,T) = e^{-r(T-t)}, F(t,T) = S_t e^{(r-q)(T-t)}$$

' Set $\widehat{C}(T_0,\widehat{K}) = max(1-\widehat{K},0)$

`/* The normalized option value with expiry` $T_0 = t$ `is the normalized payoff */`

for $i = 0;\ i < M;\ i = i+1$ **do**
  Obtain the local volatility function $\widehat{\sigma}^*(T_i,K)$ by solving problem (F.16)
  Construct $\mathbb{S}$ parametrized by $\widehat{\sigma}^*(T_i,K)$ as in equation (F.14)
  Construct $\mathbb{D}$ as in equation (F.15)
  for $j = 1;\ j < N_B;\ j = j+1$ **do**
    if $T_j^B \in (T_i, T_{i+1}]$ **then**
      Construct $\mathbb{M}(T_j^B, T_i, \widehat{\sigma}^*(T_i,.)) = (I - \mathbb{SD}(T_j^B - T_i))$
      Compute

$$\begin{bmatrix} \widehat{C}(T_j^B,\widehat{K}_0) \\ \widehat{C}(T_j^B,\widehat{K}_1) \\ \widehat{C}(T_j^B,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_j^B,\widehat{K}_N) \end{bmatrix} = \mathbb{M}^{-1}(T_j^B, T_i, \widehat{\sigma}^*(T_i,.)) \begin{bmatrix} \widehat{C}(T_i,\widehat{K}_0) \\ \widehat{C}(T_i,\widehat{K}_1) \\ \widehat{C}(T_i,\widehat{K}_2) \\ \vdots \\ \widehat{C}(T_i,\widehat{K}_N) \end{bmatrix}$$

    end
  end
end
return $\{\widehat{C}(T,\widehat{K}) | \forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

Algorithm 5: Function For Arbitrage Free Surface Construction

Function `ArbitrageFreeSurface(` t, $\mathbf{K}_{grid}$,$\mathbf{T}_{mkt}$,$optType$`)`**:**

Input: t: An option trading date t.
$\mathbf{K}_{grid} = \{K_0, K_1, \ldots, K_N\}$: A grid of strikes for outputting option value.
$\mathbf{T}_t^{mkt} = \{T_0 = t, \ldots, T_M\}$: A grid of **market** available expiries at time t.
optType: Call or Put Option.
`LVFCalibration`: The volatility Interpolation function as in Algorithm 4.

Extract $\mathbf{T}_{grid} = \{T_0^B = t, \ldots, T_{N_B}^B = T_M\}$: This grid includes every business days between t and T_M. The T_M is the maximum market expiry date in $\mathbf{T}_t^{mkt}$.

$\{\widehat{C}(T,\widehat{K})|\forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\} \leftarrow$ `LVFCalibration(`t,$\mathbf{T}_t^{mkt}$,$\mathbf{K}_{grid}$,$\mathbf{T}_{grid}$`)`

`/* Construct Put opton value using call-put parity */`

$$C_{model}^t(T,K) = \widehat{C}(T,\widehat{K})D(t,T)F(t,T), D(t,T) = e^{-r(T-t)}, F(t,T) = S_t e^{(r-q)(T-t)}$$
$$P_{model}^t(T,K) = C_{model}^t(T,K) - S_t + KD(t,T)$$

if *optType*=*Call* **then**

$$V_{model}^t(T,K) = C_{model}^t(T,K)$$

else

$$V_{model}^t(T,K) = P_{model}^t(T,K)$$

end

return $\{V_{model}^t(T,K)|\forall T \in \mathbf{T}_{grid}\ ,\ \forall K \in \mathbf{K}_{grid}\}$

other word, the total hedging horizon is $N_H = 100$ business days. We build a sequence of models to hedge those testing scenarios with the same expiry date T^{test} and different strikes $K \in \mathbf{K}^{mkt}_{grid}(t_0, T^{test})$. The testing scenario: $Scenario(T^{test}, K)$ is constructed using procedure described in Algorithm 7.

Step 3: On the initial date t_0, we prepare the training set as:

$$TrainSet = \{Scenario(T,K) | \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T), \forall T \in \mathbf{B}(T_{min}, t_0)\}$$

$$\mathbf{B}(T_{min}, t_0) = \{t | t \text{ is a business day}, T_{min} < t < t_0\}$$

where T_{min} is the earliest expiry to be included in the training set and $\mathbf{B}(T_{min}, t_0)$ is the set of all business days t with $T_{min} < t < t_0$. We set T_{min} to be 3 years prior to the intial date of the testing scenarios t_0. The grid of strikes $\mathbf{K}_{grid}(t,T)$ is defined as: $\mathbf{K}_{grid}(t,T) = \{0 = K_0 < K_1 < \cdots < 2 * K^{mkt}_{max}(t,T)\}$ with $K_i - K_{i-1} = 5, i \geq 1$ and $K^{mkt}_{max}(t,T)$ is defined as the maximum of strikes we observed in market between between t and T. In other words, we include all training hedging scenarios for which the expiry dates are before t_0 and later than T_{min}.

The validation set is:

$$ValSet = \{Scenario(T,K) | T = t_0, \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T)\}$$

In other words, we include all training hedging scenarios for which the expiry date is t_0 as the validation set. Note that for training and validation set, we do not require T and K to be observed directly from market. We train the model based on the training sets. We get the hedging position for the testing scenarios from the data-driven model for t_0 only: $\delta^M_{t_0,T,K}$.

Step 4: Similarly, on any rebalancing date $t_j > t_0$, the training set and validation set are:

$$TrainSet = \{Scenario(T,K) | \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T), \forall T \in \mathbf{B}(T_{min}, t_j)\}$$

$$\mathbf{B}(T_{min}, t_j) = \{t | t \text{ is a business day}, T_{min} < t < t_j\}$$

where $\mathbf{B}(T_{min}, t_j)$ is the set of all business days t with $T_{min} < t < t_j$.

$$ValSet = \{Scenario(T,K) | T = t_j, \forall K \in \mathbf{K}_{grid}(T - \frac{100}{250}, T)\}$$

We update the model based on the new training set. We get the hedging position from the data-driven model $\text{GRU}_{\text{TOTAL}}$ for t_j only: $\delta^M_{t_j,T,K}$. Notice that the range for the allowed expiries in the training set is extended because $\mathbf{B}(T_{min}, t_0) \subset \mathbf{B}(T_{min}, t_j)$ when $t_j > t_0$. Therefore, We have included new data into the training data set. We also validate our model based on most recent scenarios expiring on t_j.

Algorithm 6: Function For Constructing Training Hedging Scenarios With Expiry Date T

Function `TrainingScenarioGeneration(`T,*optType*`)`**:**

Input: T: an expiry date for the hedging scenarios.
optType: Call or put option.
`ArbitrageFreeSurface`: The function for surface construction as in Alogrithm 5.

Set $t_0 = T - \frac{100}{250}$: A initial date for setting up the hedging scenarios.

Extract $\mathbf{t}_B = \{t_0, \ldots, t_N = T\}$: the set of business dates between t_0 and T sorted in ascending order.

Extract $K^{mkt}_{max}(t_0, T)$: the maximum of strikes we observed in market between between t_0 and T.

Construct the grid of strikes: $\mathbf{K}_{grid}(t_0, T) = \{0 = K_0 < K_1 < \cdots < 2 * K^{mkt}_{max}(t_0, \widehat{T})\}$ where $K_i - K_{i-1} = 5$

for $t \in \mathbf{t}_B$ **do**

`/* Construct the Surface for each date` $t \in \mathbf{t}_B$ `*/`

$\mathbf{T}^{mkt}_t \leftarrow$ the set of market expiries at t

`ArbitrageFreeSurface(`t, $\mathbf{K}_{grid}(t_0, T)$,$\mathbf{T}^{mkt}_t$,optType)

end

for $t \in \mathbf{t}_B$ **do**

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

Extract underlying price S_t directly from market.

Extract option value $V_{t,T,K} = V^t_{model}(T, K)$ on the ArbitrageFreeSurface at t.

Construct a vector of features $\mathbf{y}^{T,K}_t$ based on **model** option value.

end

end

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

`/*` $\{S_t | \forall t \in \mathbf{t}_B\}$: `the time-series of underlying prices.`
$\{V_{t,T,K} | \forall t \in \mathbf{t}_B\}$:`the time-series of option prices for a hedging scenario identified by` (T, K).
$\{\mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$:`the time-series of feature vectors for a hedging scenario identified by` (T, K). `*/`

$Scenario(T, K) \leftarrow \{S_t, V_{t,T,K}, \mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$

end

return $\{Scenario(T, K) | \forall K \in \mathbf{K}_{grid}(t_0, T)\}$

Algorithm 7: Function For Constructing Testing Hedging Scenarios With Expiry Date T

Function `TestingScenarioGeneration(`T,*optType*`)`**:**

Input: T: A **market** expiry date for the hedging scenarios.
optType: Call or put option.
`ArbitrageFreeSurface`: The function for surface construction as in Alogrithm 5.

Set $t_0 = T - \frac{100}{250}$: A initial date for setting up the hedging scenarios.

Extract $\mathbf{t}_B = \{t_0, \ldots, t_N = T\}$: the set of business dates between t_0 and T sorted in ascending order.

Extract all **market** available strikes at t_0 for the expiry T as the grid of strikes: $\mathbf{K}^{mkt}_{grid}(t_0, T) = \{K^{mkt}_{t_0,T,1}, \ldots, K^{mkt}_{t_0,T,N_K}\}$

for $t \in \mathbf{t}_B$ **do**

`/* Construct the Surface for each date` $t \in \mathbf{t}_B$ `*/`

$\mathbf{T}^{mkt}_t \leftarrow$ the set of market expiries at time t

`ArbitrageFreeSurface(`t, $\mathbf{K}^{mkt}_{grid}(t_0, T)$,$\mathbf{T}^{mkt}_t$,optType)

end

for $t \in \mathbf{t}_B$ **do**

for $K \in \mathbf{K}^{mkt}_{grid}(t_0, T)$ **do**

Extract underlying price S_t directly from market.

if $V^{mkt}_{t,T,K}$ *does not exist* **then**

Extract option value as $V_{t,T,K} = V^t_{model}(T, K)$ on the ArbitrageFreeSurface at t.

Construct a vector of fetures $\mathbf{y}^{T,K}_t$ based on **model** option value.

else

Extract option value as $V_{t,T,K} = V^{mkt}_{t,T,K}$.

Construct a vector of features $\mathbf{y}^{T,K}_t$ based on **market** option value.

end

end

end

for $K \in \mathbf{K}_{grid}(t_0, T)$ **do**

`/*` $\{S_t | \forall t \in \mathbf{t}_B\}$`:` `the time-series of underlying prices.`
$\{V_{t,T,K} | \forall t \in \mathbf{t}_B\}$`:the time-series of option prices for a hedging scenario identified by` (T, K)`.`
$\{\mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$`:the time-series of feature vectors for a hedging scenario identified by` (T, K)`.` `*/`

$Scenario(T, K) \leftarrow \{S_t,\ V_{t,T,K},\ \mathbf{y}^{T,K}_t | \forall t \in \mathbf{t}_B\}$

end

return $\{Scenario(T, K) | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T)\}$

Algorithm 8: Function For Building Model To Hedge Scenarios With the Expiry Date T^{test}

Function `BuildModelForScenarios(` T^{test}, *optType*`)`**:**

Input: T^{test}: A **market** expiry date for the testing hedging scenarios.
optType: Call or put option.
`TestingScenarioGeneration`: The function for generating testing scenarios as in Algorithm 7.
`TrainingScenarioGeneration`: The function for generating training scenarios as in Algorithm 6.

Set $t_0 \leftarrow T^{test} - \frac{100}{250}$

Extract $\mathbf{t}_{RB} = \{t_0, \ldots, t_j, \ldots, t_{N_{rb}-1}\}$: the set of **rebalancing** dates sorted in ascending order. Each rebalancing time is $t_j = t_0 + j \times \Delta t$ and Δt is the gap between the adjacent rebalancing dates.

Extract all market available strikes at t_0 for the expiry T^{test} as the grid of strikes:$\mathbf{K}^{mkt}_{grid}(t_0, T^{test})$

TestSet $\leftarrow$ `TestingScenarioGeneration`$(T^{test}, \text{optType})$

Set $T_{min} \leftarrow t_0 - 3$: T_{min} is 3 years prior to t_0.

for $j = 0, j < N_{rb}, j = j+1$ **do**

  TrainSet $\leftarrow \emptyset$

  Extract $\mathbf{B}(T_{min}, t_j)$: the set of all business days between T_{min} and t_j.

  for $T \in \mathbf{B}(T_{min}, t_j)$ **do**

    TrainSubSet$\leftarrow$ `TrainingScenarioGeneration`$(T, \text{optType})$

    TrainSet $\leftarrow$ TrainSet $\cup$ TrainSubSet

  end

  ValSet$\leftarrow$ `TrainingScenarioGeneration`$(t_j, \text{optType})$

  Bulid model based on TrainSet

  Validate model based on ValSet

  Compute the hedging position at t_j for TestSet:
  $\{\delta^M_{t_j,T,K} | \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T), T = T^{test}\}$

end

return $\{\delta^M_{t,T,K} | \forall t \in \mathbf{t}_{RB}, \forall K \in \mathbf{K}^{mkt}_{grid}(t_0, T), T = T^{test}\}$