Calibration of European Option Pricing Models Using Neural Networks

Peetu Luotonen

School of Science

Thesis submitted for examination for the degree of
Master of Science in Technology.
Espoo 17.4.2023

Supervisor

Prof. Ahti Salo

Advisor

Sakari Malka M.Sc. (Tech.&Econ.)

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Peetu Luotonen

Title Calibration of European Option Pricing Models Using Neural Networks

Degree programme Mathematics and Operations Research

Major Applied Mathematics **Code of major** SCI3053

Supervisor Prof. Ahti Salo

Advisor Sakari Malka M.Sc. (Tech.&Econ.)

Date 17.4.2023 **Number of pages** 63+4 **Language** English

Abstract

An accurate modeling of option prices is important for many financial applications, such as hedging, or estimation of the underlying asset return distribution. For European options in particular, there exist many parametric pricing models. Many of these models have been developed in an attempt to improve the previous ones, for instance, by allowing a non-Gaussian distribution for the log returns of the underlying asset, or by assuming a non-constant asset volatility. The parameters of the models are selected to match the observed option market prices as well as possible, using some optimization method. This process is known as model calibration. Each option contract and time step can be calibrated separately, or the same parameter values can be used for multiple contracts and/or time steps.

In the so-called inverse map approach, one not only obtains the implied model parameters, but also a mapping from the relevant market variables to the unobservable model parameters. After the inverse map has been learned, it can be used to obtain the implied parameters and price predictions of future observations. In this thesis, the inverse map is modeled as a simple feedforward neural network, and the approach is applied to four parametric option pricing models. For calibration, the European SPX options are used. The models are evaluated using multiple metrics and time periods. Moreover, to account for the stochastic training of the neural networks, each calibration process is also repeated multiple times using different random seeds.

The overall results are promising. By using neural networks with two hidden layers and ten units per hidden layer, and by calibrating to calls and puts separately, over 50% of the out-of-sample price predictions land between the bid and ask prices of the options, in the case of all four models. Moreover, for the best version (seed) of each model, the average out-of-sample pricing error relative to the bid-ask spread is well below one. However, the results are not identical between different option maturities and strike prices, and time periods. In particular, the relative pricing errors are the largest for out of the money and short maturity options. Moreover, the models have the worst out-of-sample performance during the turbulent market year 2020.

Keywords option pricing, model calibration, neural networks, optimization, mathematical finance

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Peetu Luotonen

Työn nimi Eurooppalaisten optiohinnoittelumallien kalibrointi neuroverkkojen avulla

Koulutusohjelma Matematiikka ja operaatiotutkimus

Pääaine Sovellettu matematiikka **Pääaineen koodi** SCI3053

Työn valvoja Prof. Ahti Salo

Työn ohjaaja DI/KTM Sakari Malka

Päivämäärä 17.4.2023 **Sivumäärä** 63+4 **Kieli** Englanti

Tiivistelmä

Optioiden hintojen realistinen mallinnus on tärkeää monissa finanssialan sovelluskohteissa, kuten investointien suojaamisessa tai kohde-etuuden tuottojakauman estimoinnissa. Erityisesti eurooppalaisille optioille on kehitetty monia parametrisia hinnoittelumalleja. Monet näistä malleista pyrkivät parantamaan vanhempien mallien ominaisuuksia esimerkiksi sallimalla muuttuvan kohde-etuuden volatiliteetin tai muun kuin normaalijakauman kohteen logaritmisille tuotoille. Käytännössä mallien parametrit valitaan niin, että mallien antamat hintaestimaatit vastaavat optioiden markkinahintoja mahdollisimman hyvin. Tämä mallien kalibrointi voidaan tehdä jokaiselle optiolle ja ajanhetkelle erikseen tai usealle optiolle ja/tai ajanhetkelle samaan aikaan.

Kun käytetään niin kutsuttua käänteisfunktio-lähestymistapaa, opitaan mallien implikoimien parametrien lisäksi kuvaus markkinamuuttujista mallien parametreihin. Tämän jälkeen käänteisfunktiolla voidaan tuottaa myös tulevien optioiden hinta-arviot, ja optioiden implikoimat parametrit. Tässä työssä käänteisfunktiota mallinnetaan yksinkertaisena eteenpäin kytkettynä neuroverkkona, ja lähestymistapaa sovelletaan neljään eri optiohinnoittelumalliin. Mallit kalibroidaan eurooppalaisilla SPX-optiolla, ja mallien suorituskykyä arvioidaan eri mittareiden ja ajanjaksojen avulla. Lisäksi jokainen kalibrointi toistetaan monta kertaa, koska neuroverkkojen opetusprosessi on osittain satunnainen.

Saadut tulokset ovat yleisellä tasolla lupaavia. Kun neuroverkoissa käytetään kahta piilotettua kerrosta ja kymmenen yksikköä per kerros, ja kun mallit kalibroidaan osto- ja myyntioptioihin erikseen, yli 50% testijaksojen hintaennusteista osuu option tarjoustasojen väliin kaikkien mallien tapauksessa. Lisäksi jokaisen mallin paras satunnaisversio saavuttaa testijaksoilla osto- ja myyntitasojen väliseen etäisyyteen suhteutetun hinnoitteluvirheen, jonka arvo on selvästi alle yksi. Tulokset eivät kuitenkaan ole samanlaisia eri aikajaksojen, maturiteettien tai toteutushintojen välillä. Tarkemmin sanottuna, suhteelliset hinnoitteluvirheet ovat suurimpia korkean totetutushinnan osto-optioille, matalan toteutushinnan myyntioptioille, sekä lyhyen maturiteetin osto- ja myyntioptioille. Lisäksi mallien testitulokset ovat heikoimpia poikkeuksellisen markkinavuoden 2020 aikana.

Avainsanat optioiden hinnoittelu, mallien kalibrointi, neuroverkot, optimointi, rahoitusmatematiikka

Contents

Abstract iii

Abstract (in Finnish) iv

Contents v

Symbols and Abbreviations vi

1 Introduction 1

2 Option Pricing Theory 3
- 2.1 Basics . . . 3
- 2.2 The Black-Scholes Model . . . 5
  - 2.2.1 Risk-Neutral Valuation . . . 8
- 2.3 The Corrado-Su Model . . . 10
- 2.4 The Heston Model . . . 14
- 2.5 The Bates Model . . . 19

3 Model Calibration 22
- 3.1 Calibration Methods . . . 22
- 3.2 Neural Network Approach . . . 25
- 3.3 Model Training . . . 28
- 3.4 Implementation Considerations . . . 32

4 Numerical Results 34
- 4.1 Data . . . 34
- 4.2 Model Evaluation . . . 38
- 4.3 Model Parameters . . . 49
- 4.4 Discussion . . . 55

5 Conclusions 59

References 61

A Model Evaluation for Puts 64

Symbols and Abbreviations

Symbols

S_0	Current price of the underlying asset
K	Option strike price
T	Option time to expiration in years
r	Risk-free rate
q	Dividend yield of the underlying asset
σ	Black-Scholes and Corrado-Su volatility
μ_3	Corrado-Su skewness
μ_4	Corrado-Su kurtosis
κ	Heston mean reversion coefficient of variance
θ	Heston mean variance
V_0	Heston initial variance
ω	Heston volatility of volatility
ρ	Heston correlation between the asset price and volatility
λ_J	Bates annual jump frequency
μ_J	Bates mean relative jump size
σ_J	Bates standard deviation of logarithmic jumps

Abbreviations

ITM	In The Money
ATM	At The Money
OTM	Out of The Money
B-S	Black-Scholes
C-S	Corrado-Su
SV	Stochastic Volatility
NN	(Artificial) Neural Network
TF	TensorFlow
TFP	TensorFlow Probability

1 Introduction

Options are popular financial instruments. In 2022, over 10.3 billion total contracts were cleared by the Options Clearing Corporation, the largest equity derivatives clearing organization in the United States [OCC, 2023]. This was a change of about +4.5% from 2021, +38% from 2020, and +140% from 2017. While options have traditionally been used by large financial institutions, they have also attracted many retail investors in recent years. In addition to being great hedging instruments, options offer large potential profits due to leverage, although they are very risky for the same reason. Therefore, a good understanding of options is important for practitioners and investors alike. The mathematics of option pricing can be rather involved, even for the relatively simple European options. Still, realistic and accurate option pricing models can be valuable for not only hedging, but also risk analysis, forecasting, and other financial applications.

Starting from the work of Black and Scholes [1973], many different models have been proposed for the pricing of European options. While revolutionary at the time, and still in widespread use today, the Black-Scholes model has shortcomings. Namely, its assumptions of constant volatility and log-normal returns for the underlying asset are inconsistent with the observed market prices. In particular, stocks and stock indices tend to have short-term (logarithmic) return distributions with negative skewness and positive excess kurtosis, which the Black-Scholes model is unable to capture. To address this, many alternative models, which do not rely on these assumptions, have been developed. Examples include expansions of the normal density to allow nonzero skewness and excess kurtosis [Corrado and Su, 1996; León et al., 2009], q-Gaussian return distributions [Borland, 2002; Borland and Bouchaud, 2004], jump-diffusion models [Merton, 1976; Kou, 2002], stochastic volatility models [Hull and White, 1987; Heston, 1993], stochastic volatility models with jumps [Bates, 1996], and pure jump processes [Madan et al., 1998]. All these models have a set of parameters which cannot be directly observed, such as the volatility of the underlying asset, and these parameters can be chosen such that an optimal fit to the market option prices is obtained. This *model calibration* is typically achieved by minimizing some error, such as the mean-squared error, between the model prices and the market prices.

An alternative, and a more recent, approach is to directly obtain a fit to the market prices without the use of parametric option pricing models. In this non-parametric data-driven approach, the model attempts to 'learn' a mapping from the relevant market variables to the option prices. *Neural networks* (NN) are promising candidates for this task, because they can learn highly nonlinear relationships (which are frequent in financial data) between the features and the labels. Additionally, the availability of high-level machine learning libraries, such as TensorFlow developed by the Google Brain research team, makes

it easy to train and deploy the models. Ruf and Wang [2020] provide an extensive review of NN option pricing in the literature. In some cases, the NN models can outperform the parametric ones in terms of prediction error, but they have some drawbacks. First, the models lack some interpretability due to the black-box nature of neural networks. Second, the non-parametric models are rarely arbitrage-free, and often do not produce realistic *Greeks* (partial derivatives of the option price with respect to different variables). Specific precautions can be made [Itkin, 2019], but this may complicate the training process, and weaken the prediction performance as a result.

In this thesis, the interpretability and no-arbitrage properties of parametric option pricing models are combined with the universal approximation capabilities of neural networks. Following the framework proposed by Andreou et al. [2010], networks are used to calibrate parametric models to S&P 500 index (SPX) European options. More specifically, each network outputs the unobservable model parameters which are then fed into the corresponding pricing formula as inputs. The weights of the network are selected by minimizing some error between the pricing formula outputs and the market prices. Due to the automatic differentiation feature of neural networks, it is easy to compute the gradient of the option price with respect to the network weights, which makes the fitting process straightforward. This feature also provides an automatic computation of the Greeks. Moreover, the framework makes it possible to calibrate multiple models with relatively little extra work, as only the pricing formula needs to be changed between different models. After the calibration process, the model outputs (price predictions, parameters and Greeks) are available for further use, such as the calculation of hedging ratios and implied return distributions.

This thesis is structured as follows. Chapter 2 contains a brief introduction to options (Section 2.1) and presents the option pricing models that have been selected for use in this thesis (Sections 2.2-2.5). The sections outline the derivations of the option pricing formulas for each model and give some additional remarks. Then, Chapter 3 gives a quick rundown of neural networks and general model calibration methods, and describes the model calibration setup. Additionally, it presents details about the model training and offers considerations about the implementation in TensorFlow. Chapter 4 gives information about the data, presents the model evaluation results using different metrics, visualizes the model outputs and discusses the model limitations. Finally, Chapter 5 concludes the thesis.

2 Option Pricing Theory

2.1 Basics

An *option* is a financial contract that gives its holder the right to buy or sell a certain amount of an asset at a specific time and price. These two types of option contracts are called *calls* and *puts*, respectively. The two most popular styles of options are the American options, which give the right to exercise the option at any time before or at the expiration, and the European options, which can only be exercised at the expiration. This thesis focuses on the European options that have exact closed-form pricing formulas.

In general, the variables that affect the price of the option are the underlying asset price S_0, strike price K, time to expiration (or maturity) T, risk-free rate of return r and the dividend yield of the underlying asset q. The actual dividends of assets are discrete, but here they are approximated by a continuous yield. In addition, the option price is affected by parameters of the underlying asset distribution. An example of this is the asset *volatility* σ, when the asset *log returns* $X_t := \ln(S_t/S_0)$ are assumed to be normally distributed.

The variables above can have different effects on the option price depending on whether the option is a call or a put. For instance, as the difference between the underlying price and the strike price $(S_0 - K)$ increases, the price of a call increases, while the price of a put decreases. A call (put) option is said to be *in the money* (ITM) when $S_0 > K$ ($S_0 < K$), *at the money* (ATM) when $S_0 = K$, and *out of the money* (OTM) when $S_0 < K$ ($S_0 > K$). At the time of expiration, an out of the money (or at the money) option is worthless, while an in the money option is worth its *intrinsic value*, $S_T - K$ for calls and $K - S_T$ for puts. Here, S_T is the underlying price at the expiration time T. Figure 1 shows an example of the profit for both contract types as a function of $S_T - K$. Other variables that affect the price of calls and puts differently are the risk-free rate r and the dividend yield q. An increase in r tends to increase the value of calls and decrease the value of puts, while q tends to have the opposite effect on both contract types.

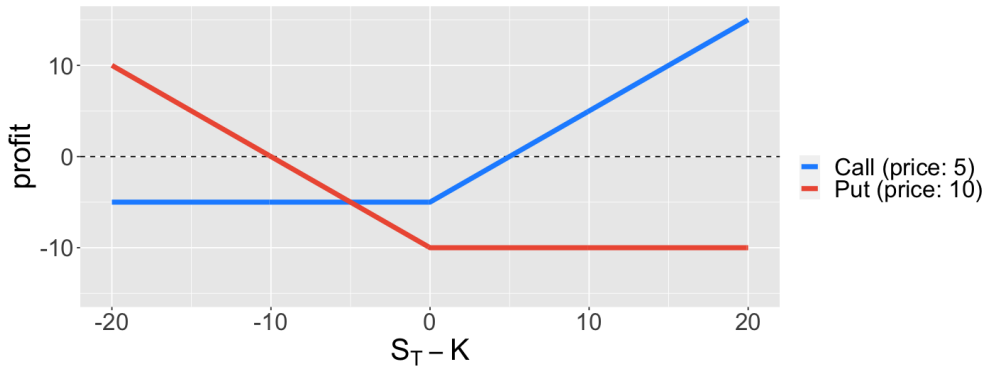

Figure 1: Profit in dollars as a function of $S_T - K$ for a call worth 5 dollars and a put worth 10 dollars.

The time to expiration T usually increases the price of both calls and puts, but this need not be the case for European options. For example, a large future dividend may make a call option expiring before the ex-dividend date more valuable than the one expiring after it. Finally, the distribution parameters tend to make both calls and puts more expensive when the distribution becomes 'wider' (e.g., when σ increases), and less expensive when the distribution is more 'narrow'. This is due to the fact that wider distributions imply the possibility of larger price changes in either direction. As option payoffs are asymmetric with unlimited upside, they are more valuable in turbulent markets. On the other hand, a skewed distribution may have a different effect on the prices of calls and puts. For instance, an underlying distribution with negative skewness increases the price of OTM puts (as this increases the likelihood of them ending up in the money), but decreases the price of OTM calls.

An important relationship between the prices of European calls and puts with the same strike price and maturity is the *put-call parity*

$$c + Ke^{-rT} = p + S_0 e^{-qT}, \tag{1}$$

where c and p are the prices of the call and put, respectively. This result can be derived by constructing two portfolios; one with the call and a risk-free asset, and the other with the put and the underlying asset, and noting that both will provide the same payoff at the time when the call and the put expire. In the absence of arbitrage, these portfolios must then be worth the same today, and the equality in Equation (1) follows. The put-call parity is a very useful property for European options, as one needs to only derive the call price c under the specific option pricing model. The price of the corresponding put should then satisfy

$$p = c + Ke^{-rT} - S_0 e^{-qT}.$$

Due to market frictions, such as bid-ask spreads and other transaction costs, this relationship does not hold exactly in practise, but Equation (1) is still a good approximation in liquid markets.

Next sections present the option pricing models used in later parts of the thesis. We start from the famous Black-Scholes model, and then move on to more complex models. Additionally, the concept of risk-neutral valuation is briefly introduced. For each model, we briefly motivate the reasons for its development, demonstrate the effect of its parameters on the underlying asset distribution, and present methods for sampling from the distribution. Finally, we give a fair price for a European call under the model assumptions. This suffices, since the corresponding put price can be recovered using the parity in Equation (1).

2.2 The Black-Scholes Model

Black and Scholes [1973] introduced the first parametric option pricing model that does not rely on subjective risk preferences of investors. The *Black-Scholes model* assumes a geometric Brownian motion for the underlying price

$$\frac{\mathrm{d}S_t}{S_t} = \mu \mathrm{d}t + \sigma \mathrm{d}W_t, \tag{2}$$

where $\mathrm{d}W_t = \epsilon\sqrt{\mathrm{d}t}$ is a Wiener process with $\epsilon \sim N(0,1)$. That is, the log return $X_t = \ln(S_t/S_0)$, where t is the duration in years, is assumed to be normally distributed with a mean μ and a standard deviation σ. The parameter σ is known as the volatility of the asset, as mentioned in Section 2.1. It then follows that the *return* of the asset R_t is log-normally distributed, that is,

$$R_t := S_t/S_0 = \exp(X_t) \sim \text{Lognormal}(\mu, \sigma^2).$$

Under the assumption of geometric Brownian motion, R_t can be sampled by directly discretizing Equation (2) as

$$\frac{\Delta S_t}{S_t} = \mu \Delta t + \sigma\epsilon\sqrt{\Delta t}, \tag{3}$$

where $\Delta S_t = S_{t+\Delta t} - S_t$, and in the limit $\Delta t \to 0$, the continuous process is recovered. However, it is advisable to sample the logarithm of the price instead, since (3) may lead to negative prices. Any deterministic function $f_t = f(S_t, t)$ of the underlying asset follows the stochastic process given by Ito's lemma [Itô, 1951]

$$\mathrm{d}f = \left(\frac{\partial f}{\partial S}\mu S + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\right)\mathrm{d}t + \frac{\partial f}{\partial S}\sigma S \mathrm{d}W, \tag{4}$$

where the subscript t has been dropped for simpler notation. Now applying (4) with $f(S,t) = \ln S$ yields

$$\ln\left(\frac{S_t}{S_0}\right) = X_t = \left(\mu - \frac{\sigma^2}{2}\right)\mathrm{d}t + \sigma \mathrm{d}W_t,$$

and thus, R_t is given by

$$R_t = \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma\epsilon\sqrt{t}\right\}. \tag{5}$$

Figure 2 illustrates *rate of returns* $R_t - 1$ that have been sampled from Equation (5) using different volatilities.

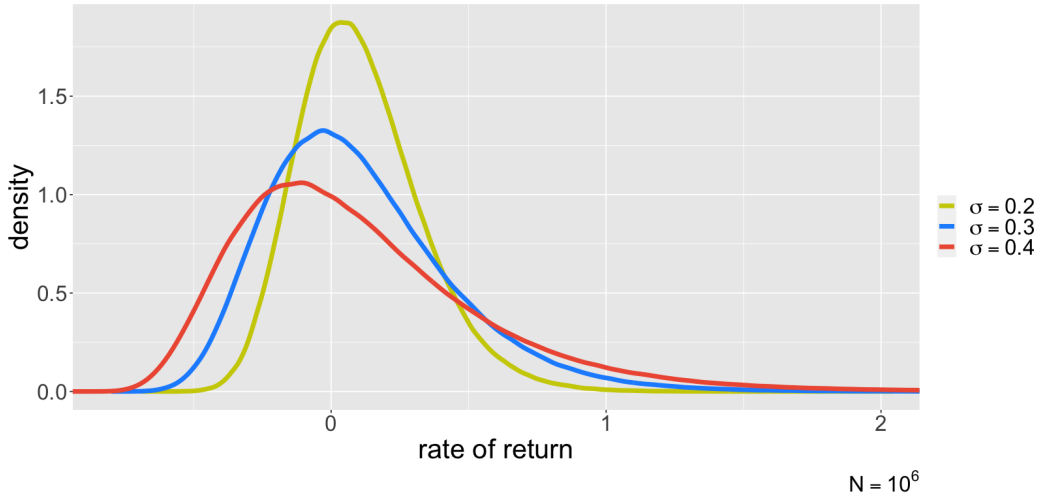

Figure 2: Yearly rate of return densities with different volatilities σ and a fixed expected (log) return $\mu = 0.1$.

We now derive a fair European call price under the Black-Scholes model. Let $f = f(S,t)$ be a function that gives the price of a financial derivative, where S is the underlying asset price. Then, f follows a stochastic process given in Equation (4). Now, a partial differential equation for f can be derived using concept of *replication*. That is, we construct a portfolio from the underlying asset and the risk-free asset such that the coefficients of the portfolio with respect to $\mathrm{d}t$ and $\mathrm{d}W$ match those of the derivative given by (4). In other words, changes in the values of the derivative and the replicating portfolio will be the same for infinitesimal changes in time and underlying price. Assuming that there are no arbitrage opportunities, the price of the derivative must then equal that of the replicating portfolio. This leads to the

Black-Scholes equation

$$\frac{\partial f}{\partial t} + \frac{\partial f}{\partial S}(r-q)S + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2 = rf. \tag{6}$$

For a European call, the boundary conditions are

$$\begin{aligned} f(S,T) &= \max(S-K,0), \\ f(0,t) &= 0. \end{aligned}$$

From these conditions, the price c of the call under the Black-Scholes model is given by [Black and Scholes, 1973]

$$c = S_0 e^{-qT} N(d_1) - K e^{-rT} N(d_2), \tag{7}$$

where

$$d_1 = \frac{\ln(S_0/K) + (r - q + \sigma^2/2)T}{\sigma\sqrt{T}}, \tag{8}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{9}$$

and N is the standard cumulative normal distribution function. By the put-call parity from Equation (1), the price of a European put is then

$$p = K e^{-rT} N(-d_2) - S_0 e^{-qT} N(-d_1). \tag{10}$$

For simplicity and analytic tractability, the Black-Scholes model assumes a log-normal distribution for the underlying asset returns. However, the prices observed in the market often contradict this assumption. In particular, log return distributions of stocks often exhibit negative skewness and excess kurtosis in short time intervals. To illustrate this, Figure 3 shows an empirical weekly log return distribution for the S&P 500 index. From the figure, it is evident that the distribution is non-Gaussian. To address the discrepancy between the empirical data and the model assumptions, the underlying price process in Equation (2) should be modified. Before that, we discuss the concept of risk-neutral valuation.

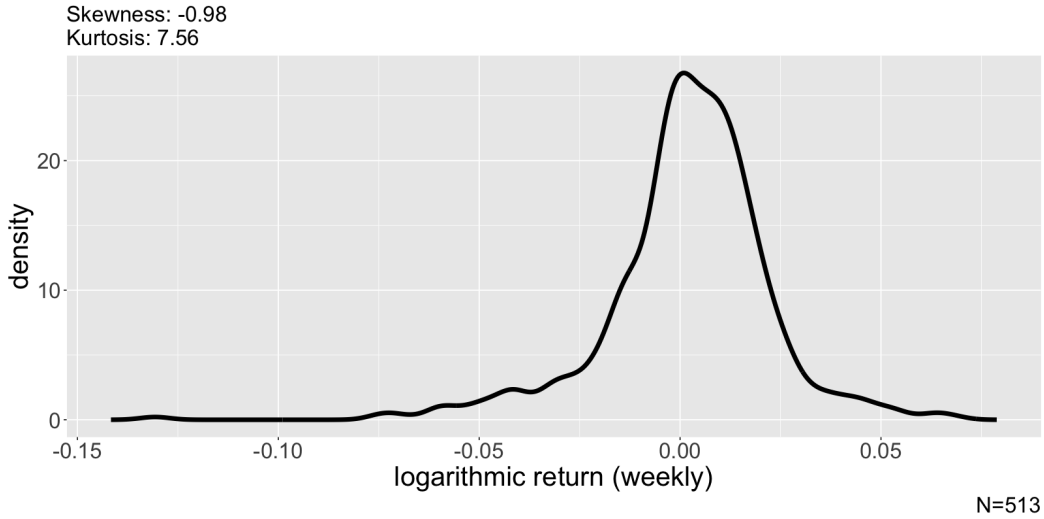

Figure 3: Weekly log return distribution of the S&P500 index between the years 2012 and 2022. The empirical distribution has negative skewness and positive excess kurtosis. The returns have been computed by taking the (log) difference between consecutive Friday closing prices.

2.2.1 Risk-Neutral Valuation

In general, the prices of European options can be calculated in two ways. The first way is to obtain a partial differential equation (such as the one in Equation (6)) via replication, and to solve it with suitable boundary conditions. The second approach is to compute the discounted expected payoff of the option, or

$$\begin{aligned} c &= e^{-rT}\mathbb{E}^{\mathbb{Q}}[\max(S_T - K, 0)] \\ &= e^{-rT}\int_K^\infty (S_T - K)p(S_T)\mathrm{d}S_T, \end{aligned} \tag{11}$$

where $\mathbb{Q}$ is a *risk-neutral measure*, or *equivalent martingale measure*, with the risk-neutral probability density function p. Under $\mathbb{Q}$, the discounted asset price $S_T e^{-(r-q)T}$ is a *martingale*, i.e., a stochastic process with zero drift term ($\mu = 0$ in Equation (2)). The technique of pricing options using Equation (11) is known as *risk-neutral valuation*.

The risk-neutral measure is unique if and only if the markets are complete, that is, an *Arrow-Debreu security* can be constructed for all possible future states. An Arrow-Debreu security for a given state is a security that pays a unit amount in that state, and nothing in all other states. The prices of these securities ('state prices') are called *risk-neutral probabilities*, since they satisfy the axioms of probability, namely non-negativity and summing up to one. However, the equivalent martingale measure $\mathbb{Q}$ is not equal to the 'true' probability measure $\mathbb{P}$ that controls the price changes of the underlying asset.

Informally, a change from $\mathbb{P}$ to $\mathbb{Q}$ is achieved by the addition of a drift term [Baxter et al., 1996; Mitra, 2011]

$$\mathrm{d}W^{\mathbb{Q}} = \mathrm{d}W^{\mathbb{P}} + \lambda(t)\mathrm{d}t, \tag{12}$$

where $W^{\mathbb{Q}}$ and $W^{\mathbb{P}}$ are Wiener processes under $\mathbb{Q}$ and $\mathbb{P}$, respectively, and $\lambda(t)$ is known as the *market price of risk*. This is known as the *Girsanov theorem*. For complete markets, only the choice $\lambda = (\mu - r)/\sigma$ leads to $S_T e^{-(r-q)T}$ being a martingale. Thus, the market price of risk (and therefore also $\mathbb{Q}$) is unique, and the expectation in (11) is unambiguously defined. The value $\lambda = (\mu - r)/\sigma$ is also called the *Sharpe ratio* [Sharpe, 1966]. For incomplete markets, there exist multiple λ (and $\mathbb{Q}$) for which the discounted asset price is a martingale. In this case, there exist multiple fair prices c corresponding to different risk-neutral measures. However, the existence of some measure $\mathbb{Q}$, along with the absence of arbitrage, allow the use of Equation (11) as if the markets were complete [Duffie et al., 2000].

Let $F_0 := S_0 e^{(r-q)T}$ be the *forward price* of the underlying asset. Equation (11) can now be expressed as

$$\begin{aligned} c &= e^{-rT}\int_K^\infty S_T p(S_T)\mathrm{d}S_T - Ke^{-rT}\int_K^\infty p(S_T)\mathrm{d}S_T, \\ &= F_0 e^{-rT}\int_K^\infty [S_T/F_0]\, p(S_T)\mathrm{d}S_T - Ke^{-rT}\int_K^\infty p(S_T)\mathrm{d}S_T \\ &= S_0 e^{-qT} P_1 - Ke^{-rT} P_2, \end{aligned} \tag{13}$$

where

$$\begin{aligned} P_1 &= \int_K^\infty [S_T/F_0]\, p(S_T)\mathrm{d}S_T, \\ P_2 &= \int_K^\infty p(S_T)\mathrm{d}S_T. \end{aligned}$$

This is a general formula for the price of a European call, where the risk-neutral probabilities $P_1, P_2 \in [0, 1]$ depend on the risk-neutral measure $\mathbb{Q}$. The property $P_1 \in [0, 1]$ follows from the fact that S_T/F_0 is always nonnegative, and

$$P_1 = \frac{1}{F_0}\int_K^\infty S_T p(S_T)\mathrm{d}S_T \leq \frac{1}{F_0}\int_0^\infty S_T p(S_T)\mathrm{d}S_T = \frac{\mathbb{E}^{\mathbb{Q}}[S_T]}{F_0} = 1.$$

From Equation (7), it can be seen that $P_1 = N(d_1)$ and $P_2 = N(d_2)$ under the Black-Scholes model.

2.3 The Corrado-Su Model

To relax the Black-Scholes assumption of normally distributed log returns, many models have been proposed in the literature. One way to obtain non-Gaussian distributions is to expand the normal density to allow non-zero skewness and excess kurtosis for $\ln(S_t/S_0)$. A popular example is the Gram-Charlier type A series expansion

$$f(x) = \sum_{n=0}^{\infty} c_n He_n(x)\phi(x), \tag{14}$$

where ϕ is a normal density function, He_n are (probabilist's) *Hermite polynomials*

$$He_n(x) = \left(x - \frac{\mathrm{d}}{\mathrm{d}x}\right)^n,$$

and c_n are the moments of the cumulative distribution function $F(x) = \int_{-\infty}^{x} f(x)\mathrm{d}x$. In practice, it is common to only include the terms in (14) up to $n = 4$. Using this series truncation, Corrado and Su [1996] obtain the risk-neutral density for the standardized log returns

$$g(z) = n(z)\left[1 + \frac{\mu_3}{3!}He_3(z) + \frac{\mu_4 - 3}{4!}He_4(z)\right], \tag{15}$$

where μ_3 and μ_4 are the standardized skewness and kurtosis for $\ln(S_t/S_0)$, respectively,

$$z = \frac{\ln(S_t/S_0) - (r + q - \sigma^2/2)T}{\sigma\sqrt{T}} \tag{16}$$

is the standardized log return with $\mathbb{E}(z) = 0$, $\mathbb{E}(z^2) = 1$, $E(z^3) = \mu_3$ and $E(z^4) = \mu_4$,

$$n(z) = (2\pi)^{-1/2} e^{-z^2/2}$$

is the standard normal density, and $He_3(z) = z^3 - 3z$ and $He_4(z) = z^4 - 6z^2 + 3$ are the 3rd and 4th Hermite polynomials, respectively.

To obtain log return samples under the Gram-Charlier expansion, one can draw samples from the density in (15), and then transform them into $\ln(S_t/S_0)$ using Equation (16). The sampling from the density $g(z)$ can be achieved using techniques, such as *rejection sampling*. However, in this thesis, a simple discretization of $g(z)$ is used instead. More specifically, we compute the density d_i at n_B equally spaced points p_i in the range $[-5, 5]$, and calculate the empirical cumulative values $C_0 = 0$ and $C_i = \left(\sum_{j=1}^{i} d_i\right) / \left(\sum_{j=1}^{n_B} d_i\right)$ for $i = 1, ..., n_B$. Then, we draw n_s samples from a uniform distribution on the

unit interval, and assign each sample $u \in [0, 1]$ to the bin $j \in \{1, ..., n_B\}$ such that $C_{j-1} \leq u < C_j$. Finally, for each j, we obtain the sampled point z by uniformly sampling between the points p_{j-1} and p_j. Figure 4 shows log returns that are sampled using this method, with different values for μ_3 and μ_4.

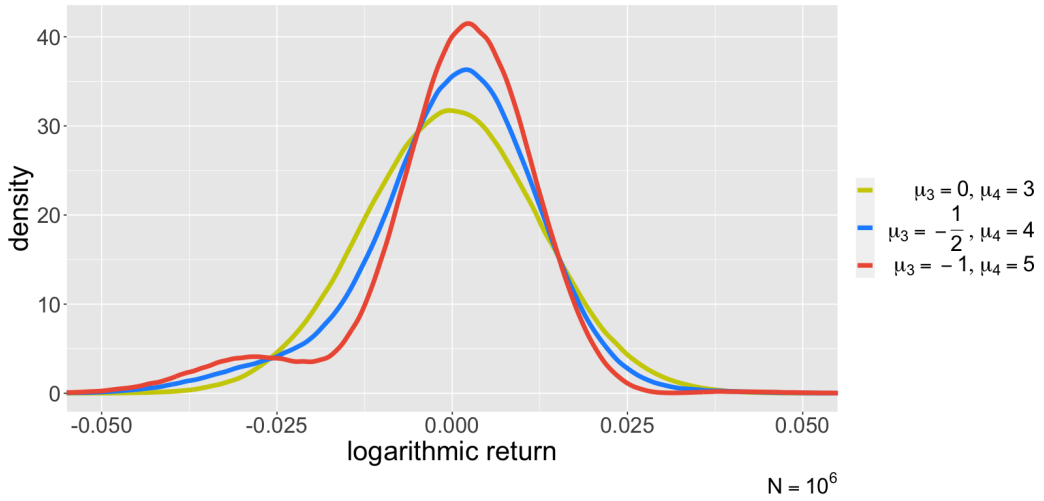

Figure 4: Empirical densities for log returns $\ln(S_t/S_0)$ with different values for the skewness μ_3 and the kurtosis μ_4. Here $r = 0, \sigma = 0.2$ and $T = 1/252$ (approximately one trading day). The density in yellow corresponds to a normal distribution, i.e., the Black-Scholes model.

Using the risk-neutral density $g(z)$ in Equation (15), a fair value for a European call can be obtained using the general risk-neutral valuation formula from Equation (11). Corrado and Su [1996] and Brown and Robinson [2002] show that the price of the call is given by

$$c = c_{BS} + \mu_3 Q_3 + \mu_4 Q_4, \tag{17}$$

where c_{BS} is the Black-Scholes call price given by Equation (7), and

$$Q_3 = \frac{1}{6} S_0 \sigma \sqrt{T} \left[P_3(d) n(d) + \sigma^2 T N(d) \right],$$
$$Q_4 = \frac{1}{24} S_0 \sigma \sqrt{T} \left[P_4(d) n(d) + \sigma^3 T^{3/2} \right],$$

where $d = d_1$ from Equation (8), and

$$P_3(x) = 2\sigma\sqrt{T} - x,$$
$$P_4(x) = x^2 - 3x\sigma\sqrt{T} + 3\sigma^2 T - 1.$$

By using the no-arbitrage condition $\mathbb{E}^{\mathbb{Q}}[S_T] = F_0 = S_0 e^{(r-q)T}$ for the risk-

neutral measure $\mathbb{Q}$ under the assumption that $g(z)$ is the probability density for the standardized log returns, Jurczenko et al. [2004] provide a modification to Equation (17), and obtain

$$c = c^*_{BS} + \mu_3 Q^*_3 + \mu_4 Q^*_4, \tag{18}$$

where

$$c^*_{BS} = S_0 e^{-qT} N(d^*) - K e^{-rT} N(d^* - \sigma\sqrt{t})$$

is the Black-Scholes call price evaluated at

$$d^* = \frac{\ln(S_0/K) + (r - q + \sigma^2/2)T - \ln(1+\omega)}{\sigma\sqrt{T}},$$

and

$$Q^*_3 = [6(1+\omega)]^{-1} S_0 \sigma\sqrt{T} P_3(d^*) n(d^*),$$
$$Q^*_4 = [24(1+\omega)]^{-1} S_0 \sigma\sqrt{T} P_4(d^*) n(d^*),$$

with

$$\omega = \frac{\mu_3}{6}\sigma^3 T^{3/2} + \frac{\mu_4 - 3}{24}\sigma^4 T^2.$$

Since the formula in Equation (18) satisfies the no-arbitrage condition, we use it instead of the formula in Equation (17), although the two formulas give very similar values in most situations [Jurczenko et al., 2004]. In this thesis, the model that prices calls according to Equation (18) is referred to as the *Corrado-Su model*.

We note that special care must be taken when using the density in (15), since $g(z)$ may exhibit multimodality and/or negative values for some pairs (μ_3, μ_4). The latter property is especially problematic, for the density would not define any probabilty distribution. To avoid this, the polynomial

$$p_4(z) := 1 + \frac{\mu_3}{3!} He_3(z) + \frac{\mu_4 - 3}{4!} He_4(z)$$

can be restricted to positive values, so that $g(z) = n(z)p_4(z) > 0$ for all z. Jondeau and Rockinger [2001] derive the equations

$$1 + \frac{\mu_3}{6} He_3(z) + \frac{\mu_4 - 3}{24} He_4(z) = 0, \tag{19}$$
$$\frac{\mu_3}{2} He_2(z) + \frac{\mu_4 - 3}{6} He_3(z) = 0, \tag{20}$$

that define the boundary (or the *envelope*) for which $p_4(z)$ is zero for a given

z. The set $D \subset \mathbb{R}^2$ delimited by this boundary contains the pairs (μ_3, μ_4) for which $g(z)$ is positive for all z. The boundary is drawn on the (μ_3, μ_4)-plane in Figure 5.

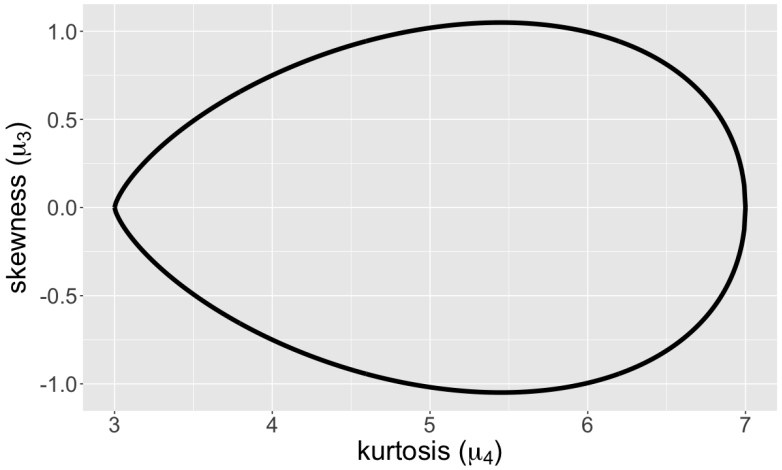

Figure 5: The boundary for which the polynomial $p_4(z)$ is zero for different $|z| \geq \sqrt{3}$. The skewness μ_3 and kurtosis μ_4 corresponding to each z are solved using equations (19) and (20). The area delimited by the boundary contains all pairs of μ_3 and μ_4 for which the density (15) is positive for all z.

Jondeau and Rockinger [2001] also present an ad-hoc method for mapping the unconstrained $(\tilde{\mu}_3, \tilde{\mu}_4) \in \mathbb{R}^2$ into constrained $(\mu_3, \mu_4) \in D$ by

$$\begin{aligned} \mu_4 &= f(\tilde{\mu}_4, 3, 7), \\ \mu_3 &= f(\tilde{\mu}_3, -s(\tilde{\mu}_4), s(\tilde{\mu}_4)), \end{aligned} \tag{21}$$

where

$$f(x; a, b) = a + (b - a)(1 + e^{-x})^{-1} \tag{22}$$

is the scaled sigmoid function, and

$$s(k) = a_j + b_j(k - 3),$$

with

$$\begin{aligned} a_i &= \frac{s_i(k_{i+1} - 3) - (k_i - 3)s_{i+1}}{k_{i+1} - k_i}, \\ b_i &= \frac{s_{i+1} - s_i}{k_{i+1} - k_i}, \end{aligned}$$

where k_i and s_i form a fine grid of kurtosis and skewness for $i = 1, ..., N$, respectively. The index j for each $k = \tilde{\mu}_4$ is selected such that $k_j < k \leq k_{j+1}$.

For each k_i, the corresponding s_i can be determined by solving the equations (19) and (20) for $\mu_3 = s_i$ and $|z| \geq \sqrt{3}$ with a fixed $\mu_4 = k_i$. The bisection algorithm is suitable for solving these equations.

2.4 The Heston Model

In the previous section, the Gram-Charlier expansion was used to add nonzero skewness and excess kurtosis to the underlying asset distribution, thus resulting in the Corrado-Su model. However, both the Black-Scholes model and the Corrado-Su model assume constant volatility for the underlying asset, which is rarely the case for stocks and stock indices. In fact, empirical volatilities tend to exhibit clustering and mean reversion, as seen in an example given in Figure 6.

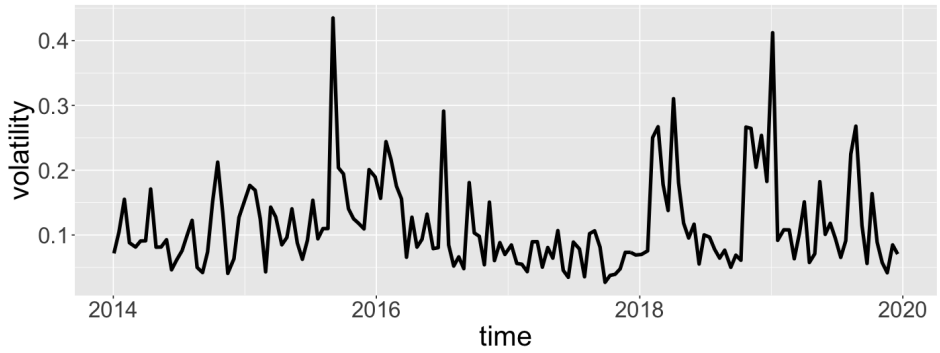

Figure 6: 10-day (non-overlapping) moving volatility of daily S&P500 prices between 2015 and 2019. The volatilities tend to cluster due to the autocorrelation between consecutive absolute price changes (large changes tend to follow large changes, and vice versa). Additionally, the volatilities tend to revert back to their long-term average.

When the volatility of an asset is modeled as a separate stochastic process, we obtain the popular class of *stochastic volatility models*. Perhaps the most widely used stochastic volatility model is the *Heston model* [Heston, 1993]

$$\mathrm{d}S = \mu S \mathrm{d}t + \sqrt{V} S \mathrm{d}W^{(1)}, \tag{23}$$

$$\mathrm{d}V = \kappa(\theta - V)\mathrm{d}t + \omega\sqrt{V}\mathrm{d}W^{(2)}, \tag{24}$$

$$\mathrm{d}W^{(1)}\mathrm{d}W^{(2)} = \rho \mathrm{d}t, \tag{25}$$

where $V = \sigma^2$ is the variance of the underlying asset, θ is the long-term variance, κ is the mean-reversion coefficient indicating how fast V reverses to θ, ω is the volatility of volatility, and ρ is the correlation between the Wiener processes $W^{(1)}$ and $W^{(2)}$. Additionally, there exists an additional parameter,

namely the initial variance V_0. The parameters κ, θ and V_0 control the overall level of volatility, negative (positive) correlation ρ leads to a return distribution with negative (positive) skewness, and an increase (decrease) of ω increases (decreases) the kurtosis of the distribution. The Heston model stands out from other stochastic volatility models, since there exists an analytical solution for European options such that the correlation between the asset price and the volatility is taken into account (see Mitra [2011] for a review of different volatility models).

When sampling from the process (23), it is advisable to sample $\ln S$ instead of S by using the discretization

$$\ln S_{t+\Delta t} = \ln S_t + (\mu - V_t/2)\Delta t + \sqrt{V_t}\Delta W_t^{(1)}. \tag{26}$$

Moreover, the variance V in process (24) can become negative when discrete time steps are used, and these negative values have to be fixed in some way. Common fixes are the *absorption*, where negative variances are set to zero, and *reflection*, where negative variances are set to their opposites [Lord et al., 2010]. These two methods can also be combined. Lord et al. [2010] show that the *full truncation* scheme

$$\begin{aligned} \tilde{V}_{t+\Delta t} &= \tilde{V}_t + \kappa(|\tilde{V}_t|^+ - \theta)\Delta t + \omega\sqrt{|\tilde{V}_t|^+}\Delta W^{(2)}, \\ V_{t+\Delta t} &= |\tilde{V}_{t+\Delta t}|^+, \end{aligned} \tag{27}$$

where $|\cdot|^+ = \max(\cdot, 0)$ and $\tilde{V}_0 = V_0$, gives the smallest bias among different simulation schemes. Using (26) and (27), one can then sample asset prices from the processes (23)-(24). The Wiener process samples are given by

$$\begin{aligned} \Delta W^{(1)} &= \epsilon_1\sqrt{\Delta t}, \\ \Delta W^{(2)} &= \left(\rho\epsilon_1 + \sqrt{1-\rho^2}\epsilon_2\right)\sqrt{\Delta t}, \end{aligned}$$

where $\epsilon_1, \epsilon_2 \sim N(0,1)$ for independent ϵ_1 and ϵ_2. Figure 7 shows log return samples with different parameters.

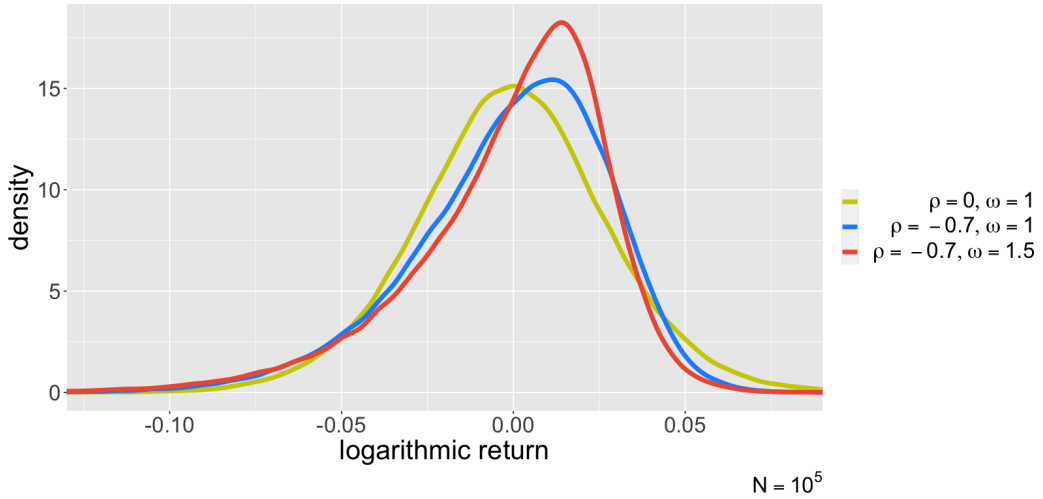

Figure 7: Empirical densities for log returns $\ln(S_t/S_0)$ under the Heston model with different values of ρ and ω. The samples were drawn with constants $\mu = 0, V_0 = \theta = 0.04$, $\kappa = 1$, $t = 5/252$ (approx. 5 trading days) and $\Delta t = t/200$ (approx. 10 trading minutes). The samples show the effect of ρ and ω on the shape of the distribution.

The risk-neutral versions of the processes in (23)-(24) are given by

$$\mathrm{d}S = rS\mathrm{d}t + \sqrt{V}S\mathrm{d}W^{(1)}, \tag{28}$$

$$\mathrm{d}V = [\kappa(\theta - V) + \lambda(S, V, t)]\,\mathrm{d}t + \omega\sqrt{V}\mathrm{d}W^{(2)}, \tag{29}$$

where $\lambda(S, V, t)$ represents the price of volatility risk (analogous to the market price of risk from Section 2.2.1). Due the second source of randomness introduced by $W^{(2)}$, the markets are incomplete under the model, so there is no unique risk-neutral measure. Thus, a suitable price of volatility risk must be chosen. On economic justifications, Heston [1993] chooses $\lambda(S, V, t) = \lambda V$ for a constant λ. Bollerslev et al. [2011] show that λ can be estimated as

$$\lambda = \gamma\rho\sigma, \tag{30}$$

where $\gamma \neq 1$ is the risk aversion coefficient of the isoelastic utility function

$$u(x) = \frac{x^{1-\gamma} - 1}{1 - \gamma}. \tag{31}$$

By change of variables

$$\begin{aligned} \kappa^* &= \kappa + \lambda, \\ \theta^* &= \kappa\theta/(\kappa + \lambda), \end{aligned} \tag{32}$$

(28)-(29) can be written in the same form as (23)-(24):

$$\mathrm{d}S = rS\mathrm{d}t + \sqrt{V}S\mathrm{d}W^{(1)}, \tag{33}$$
$$\mathrm{d}V = [\kappa^*(\theta^* - V)]\,\mathrm{d}t + \omega\sqrt{V}\mathrm{d}W^{(2)}. \tag{34}$$

From now on, we denote $\kappa = \kappa^*$ and $\theta = \theta^*$ for simpler notation, but the risk-neutral parameters can always be transformed from and to the real parameters using the equations in (32).

From (33), (34) and (25), the partial differential equation for the derivative price f can again be derived using the replication method from Section 2.2. However, now one must also consider the sensitivity of f to the asset variance V. The result is [Heston, 1993]

$$\begin{aligned}&\frac{1}{2}VS^2\frac{\partial^2 f}{\partial S^2} + \rho\omega VS\frac{\partial^2 f}{\partial S\partial V} + \frac{1}{2}\omega^2 V\frac{\partial^2 f}{\partial V^2}\\ &+ (r-q)S\frac{\partial f}{\partial S} + \kappa(\theta - V)\frac{\partial f}{\partial V} + \frac{\partial f}{\partial t} = rf.\end{aligned} \tag{35}$$

By guessing a solution f of the same form as in Equation (13), Heston [1993] derives the price of the call option from (35) by applying a Fourier inversion to the characteristic functions of the risk-neutral probabilities P_1 and P_2, and obtains

$$c = S_0 e^{-qT}P_1 - Ke^{-rT}P_2, \tag{36}$$

where

$$P_j = \frac{1}{2} + \frac{1}{\pi}\int_0^\infty f_j(u)\mathrm{d}u \tag{37}$$

for $i = 1, 2$. The functions f_j are given by

$$\begin{aligned}f_1(u) &= \mathrm{Re}\left[\frac{e^{-iu\ln(K)}\phi(u-i)}{iuF_0}\right],\\ f_2(u) &= \mathrm{Re}\left[\frac{e^{-iu\ln(K)}\phi(u)}{iu}\right],\end{aligned} \tag{38}$$

where $F_0 = S_0 e^{(r-q)T}$ is the forward price of the underlying asset, as before. The characteristic function ϕ is given by [Cui et al., 2017]

$$\begin{aligned}\phi(u) = \exp\Bigg\{&iu(\ln S_0 + rT) + \frac{\kappa\theta}{\omega^2}\left[(\varepsilon + d)T - 2\ln\left(\frac{1 - g_1 e^{dt}}{1 - g_1}\right)\right]\\ &+ \frac{V_0}{\omega^2}(\varepsilon + d)\frac{1 - e^{dt}}{1 - g_1 e^{dt}}\Bigg\},\end{aligned} \tag{39}$$

with

$$\begin{aligned}\varepsilon &= \kappa - \omega\rho iu,\\ d &= \sqrt{\varepsilon^2 + \omega^2(u^2 + iu)},\\ g_1 &= \frac{\varepsilon + d}{\varepsilon - d}.\end{aligned}$$

The integrals in Equation (37) cannot be simplified, but they can be evaluated numerically. In this thesis, we use the Gauss-Legendre quadrature

$$\int_0^1 f(x)\mathrm{d}x \approx \sum_{i=1}^{n} w_i f\left(\frac{x_i + 1}{2}\right), \tag{40}$$

where x_i is the i:th root of the *Legendre polynomial* $P_n(x)$ of degree n, and w_i is given by

$$w_i = \frac{1}{(1 - x_i^2)[P_n'(x_i)]^2},$$

for $i = 1, ..., n$. To use this approximation, the limits of the integration in Equation (37) should be changed to the unit interval. By using the change of variable $u(x) = -\ln x/C_\infty$ with [Kahl and Jäckel, 2005]

$$C_\infty = \frac{\sqrt{1 - \rho^2}}{\omega}(V_0 + \kappa\theta T),$$

Equation (36) can transformed to

$$c = e^{-rT}\int_0^1 \left[\frac{1}{2}(F_0 - K) + \frac{F_0 f_1(u(x)) - K f_2(u(x))}{\pi C_\infty x}\right]\mathrm{d}x \tag{41}$$

which can now be approximated by Equation (40).

Kahl and Jäckel [2005] note that evaluating the functions in Equation (38) is not straightforward, since the characteristic function ϕ (Equation (39)) can contain discontinuities in its current form. Moreover, an analytical gradient $\nabla_{\boldsymbol{\theta}} c$ with respect to the parameter vector $\boldsymbol{\theta} = (\kappa, \theta, V_0, \omega, \rho)$ is not available when using the representation in (39) [Cui et al., 2017]. To fix these issues, Cui et al. [2017] provide an alternative representation

$$\phi(u) = \exp\left\{iu(\ln S_0 + rT) - \frac{T\kappa\theta\rho iu}{\omega} - V_0 A + \frac{2\kappa\theta}{\omega^2}D\right\}, \tag{42}$$

where

$$A = \frac{(u^2 + iu)\sinh(dt/2)}{d + \cosh(dt/2) + \varepsilon \sinh(dt/2)}, \tag{43}$$

$$B = \ln d + \frac{(\kappa - d)T}{2} - \ln\left(\frac{d+\varepsilon}{2} + \frac{d-\varepsilon}{2}e^{-dt}\right).$$

Due to the more attractive properties, we use the representation in (42) instead of the one in (39).

2.5 The Bates Model

Due to the stochastic volatility component, the Heston model is more flexible than the Black-Scholes model, and is able to obtain a better fit to the market prices across multiple maturities and strikes. However, even the Heston model is insufficient in some situations. Namely, the diffusion processes (23)-(25) are unable to explain option prices that imply the possibility of very large price changes in a very short amount of time. To explain these sudden moves, one can add *jumps* to the underlying asset process. An extension of the Heston model that introduces these jumps is the *Bates model* [Bates, 1996]

$$\frac{\mathrm{d}S}{S} = (\mu - \lambda_J \mu_J)\mathrm{d}t + \sqrt{V}\mathrm{d}W^{(1)} + k\mathrm{d}Q, \tag{44}$$

$$\mathrm{d}V = \kappa(\theta - V)\mathrm{d}t + \omega\sqrt{V}\mathrm{d}W^{(2)}, \tag{45}$$

$$\mathrm{d}W^{(1)}\mathrm{d}W^{(2)} = \rho \mathrm{d}t,$$

$$\mathbb{P}(\mathrm{d}Q = 1) = \lambda_J \mathrm{d}t. \tag{46}$$

Here λ_J is the annual frequency of jumps, μ_J is the mean relative jump size, k is the random (conditional) jump size with a log-normal distribution

$$\ln(1+k) \sim N(\ln(1+\mu_J) - \sigma_J^2, \sigma_J^2), \tag{47}$$

with the standard deviation σ_J, and Q is a Poisson random variable with the rate parameter λ_J. In total, the Bates model contains 8 parameters: the 5 Heston parameters plus λ_J, μ_J and σ_J. With the addition of the jump term, the model is able to fit short-term OTM options better than the models with only a diffusion component, since these options often imply the possibility of large price changes in short amount of time. Negative (positive) jumps lead to a return distribution with more negative (positive) skewness. Moreover, a large jump component tends to increase the kurtosis of the distribution.

Due to the added jump component $k\mathrm{d}Q$, it is no longer straightforward to sample $\ln S$ as in the case of the Heston model. Thus, we use a direct

disretization of (44):

$$S_{t+\Delta t} = S_t[1 + (\mu - \lambda_J \mu_J)\Delta t + \sqrt{V_t}\Delta W_t^{(1)} + k_t \Delta Q_t], \tag{48}$$

where k_t is sampled from (47), and ΔQ_t is sampled from a Poisson distribution with a rate parameter $\lambda_J \Delta t$. On the other hand, since the variance V is independent of the jumps, it can still be sampled using the full truncation scheme in Equation (27). Figure 8 shows samples of $\ln(S_t/S_0)$ with different parameters.

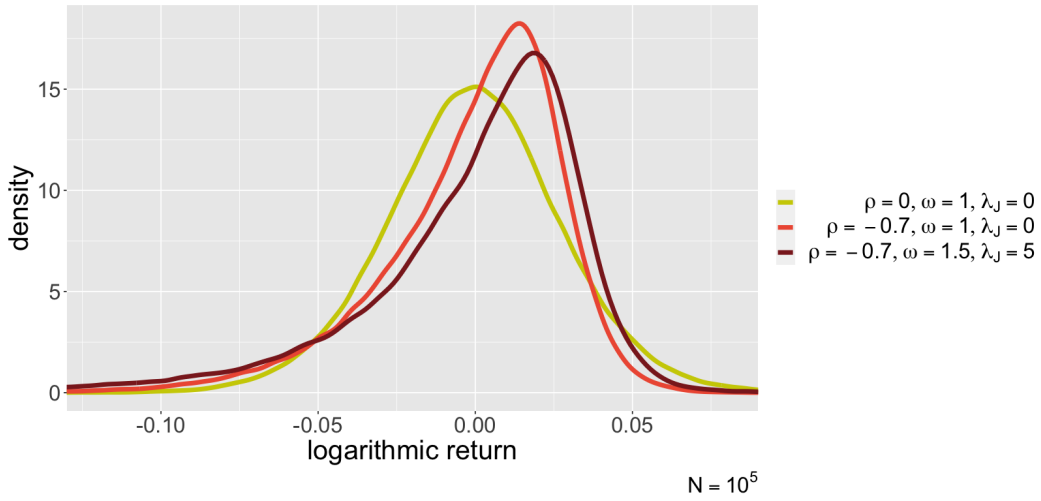

Figure 8: Empirical densities for log returns $\ln(S_t/S_0)$ under the Bates model with different values of ρ, ω and λ_J. The cases where $\lambda_J = 0$ (yellow and red) correspond to the densities of same colors in Figure 7, since a Bates model without jumps is equivalent to the Heston model. The samples were drawn with constants $\mu = 0, V_0 = \theta = 0.04$, $\kappa = 1$, $\mu_J = -0.05$, $\sigma_J = 0.05$, $t = 5/253$ (approx. 5 trading days) and $\Delta t = t/200$ (approx. 10 trading minutes). The case $\lambda_J = 5$ (brown) shows the effect of the (negative) jumps on the return distribution.

The risk-neutral version of the processes (44) and (46) are given by

$$\frac{\mathrm{d}S}{S} = (\mu - \lambda_J^* \mu_J^*)\mathrm{d}t + \sqrt{V}\mathrm{d}W^{(1)} + k^* \mathrm{d}Q^*, \tag{49}$$

$$\mathbb{P}(\mathrm{d}Q^* = 1) = \lambda_J^* \mathrm{d}t, \tag{50}$$

where k^* is distributed as

$$\ln(1 + k^*) \sim N(\ln(1 + \mu_J^*) - \sigma_J^2/2, \sigma_J^2)$$

(i.e., with the same variance σ^2 as the real jumps), and $\mathrm{d}Q^*$ has a rate

parameter λ_J^*. In addition, the risk-neutral process for V is again given by (34). The risk-neutral parameters λ_J^* and μ_J^* can be estimated as [Bates, 1996]

$$\begin{aligned} \lambda_J^* &= \lambda_J \mathbb{E}\left[1 + u_J\right], \\ \mu_J^* &= \mu_J + \frac{\mathrm{Cov}(k, u_J)}{\mathbb{E}\left[1 + u_J\right]}, \end{aligned} \tag{51}$$

where u_J is a relative change in the marginal utility function $u_x(x) := \frac{\mathrm{d}u(x)}{\mathrm{d}x}$ when a jump of relative size k occurs. For the isoelastic utility in Equation (31), the marginal utility is $u_x(x) = x^{-\gamma}$, so

$$u_J = \frac{[x(1+k)]^{-\gamma}}{x^{-\gamma}} - 1 = (1+k)^{-\gamma} - 1.$$

When converting the risk-neutral parameters from and to the real parameters using Equation (51), numerical (sampled) estimates of $\mathbb{E}[1+u_J]$ and $\mathrm{Cov}(k, u_J)$ can be used.

The call price solutions to the Heston model and the Bates model can be computed in a very similar fashion, because the Fourier inversion technique (Equation (37)) in the former can also be applied to the latter [Bates, 1996]. In fact, one only needs to modify the characteristic function $\phi(u)$; all the other parts remain the same. Thus, a fair call price under the Bates model is given by

$$c = e^{-rT} \int_0^1 \left[\frac{1}{2}(F_0 - K) + \frac{F_0 f_1^B(u(x)) - K f_2^B(u(x))}{\pi C_\infty x} \right] \mathrm{d}x, \tag{52}$$

with

$$\begin{aligned} f_1^B(u) &= \mathrm{Re}\left[\frac{e^{-iu\ln(K)} \phi_B(u - i)}{iuF_0} \right], \\ f_2^B(u) &= \mathrm{Re}\left[\frac{e^{-iu\ln(K)} \phi_B(u)}{iu} \right]. \end{aligned}$$

Here, the modified characteristic function $\phi_B(u)$ is [Date and Islyaev, 2015]

$$\phi_B(u) = \phi_H(u) \exp\left\{E(u)T\right\}, \tag{53}$$

where

$$E(u) = \lambda_J \left[\exp\left(iu\alpha_J - u^2\sigma_J^2/2 \right) - 1 \right] - \lambda_J iu \left[\exp\left(\alpha_J + \sigma_J^2/2 \right) - 1 \right],$$

with $\alpha_J = \ln(1 + \mu_J) - \sigma_J^2/2$, and $\phi_H(u)$ is the Heston characteristic function from Equation (42).

3 Model Calibration

3.1 Calibration Methods

Option pricing models, such as the ones presented in Sections 2.2-2.5, attempt to explain market option prices by assuming certain asset price dynamics, from which a fair option value can be derived using either the generalized Black-Scholes equation or risk-neutral valuation. Many of these models have been proposed in an attempt to improve the previous ones, either by lifting some unrealistic restrictions (e.g. normality of log returns or constant volatility), or by adding new features (e.g. jumps). In doing so, the models become increasingly complex, but also more flexible. However, even the most complex models have shortcomings, and this is evident by the fact that no model can produce perfectly accurate price estimates in all market situations with a fixed (time independent) set of parameters. Still, the models can provide valuable insight about the option markets.

It is possible to only consider option pricing models in the forward direction (set model parameters, then predict the prices), but arguably a more interesting consideration is the reverse backward direction (observe market prices, then infer the model parameters). In the latter approach, one can investigate what the market prices tell about the underlying asset dynamics, or more specifically, what are the market expectations for these dynamics. By doing this for multiple time steps, one can also examine how these expectations change with time. Of course, this inference is done under the hypothesis that a given model is correct, which is never truly the case, but some models can be fairly good approximations to the true market dynamics. The process of obtaining the model parameters from the market prices is called *model calibration*, and the obtained parameters are called *implied parameters* (because, as the name suggests, they are implied by the current market prices).

The goal of model calibration is to obtain the parameters which give the best possible fit to the market prices. One approach is to find the parameters separately for each available option contract such that the model prices exactly match the market prices. This can be achieved using some root-finding algorithm. Let $\boldsymbol{\theta} \in S_{\boldsymbol{\theta}}$ be a m-dimensional parameter vector of the pricing model, where $S_{\boldsymbol{\theta}} \subseteq \mathbb{R}^m$ is the set of allowed parameter combinations. Table 1 lists the parameter vector $\boldsymbol{\theta}$ for different models. Mathematically, the goal is to find the set of parameters $\boldsymbol{\theta}_i^* \in S_{\boldsymbol{\theta}}$ such that

$$f(x_i; \boldsymbol{\theta}_i^*) = y_i, \quad \forall i \in [n], \tag{54}$$

where n is the number of observations (different option contracts and time steps), $[n] := \{1, ..., n\}$, y_i are the market prices of the observations for $i \in [n]$, f is the pricing function, and $x_i \in S_x \subseteq \mathbb{R}^k$, $i \in [n]$, are vectors of other variables required by the pricing function. For European options, the variable

vector is typically

$$x = (S_0, K, T, r, q, b) \in S_x, \tag{55}$$

where b is a binary variable indicating whether the contract is a call or a put. However, this need not be the case. For instance, q can be dropped when the underlying asset does not pay dividends, b can be omitted if only calls or puts are considered, or some parameters in $\boldsymbol{\theta}$ can be fixed and included in x as constants (here $\boldsymbol{\theta}$ contains only parameters that are varied across different contracts or time steps).

Table 1: Set of parameters for different option pricing models.

Model	$\boldsymbol{\theta}$
Black-Scholes	σ
Corrado-Su	σ, μ_3, μ_4
Heston	$\kappa, \theta, V_0, \omega, \rho$
Bates	$\kappa, \theta, V_0, \omega, \rho, \lambda_J, \mu_J, \sigma_J$

Under the Black-Scholes model in particular, we have $\boldsymbol{\theta}_i = (\sigma_i)$, where $\sigma_i \in \mathbb{R}_+$ is the *implied volatility* given by the i:th observation. When plugged back into the pricing formula in Equation (7) (or Equation (10) in the case of puts), this volatility produces a price equal to the market price. Since there is only a single parameter to be calibrated in this case, $\boldsymbol{\theta}_i^*$ can be found for each contract and time step using a simple line search method, such as the bisection method. When σ_i^* are solved for multiple strikes and expirations, we obtain the so-called *implied volatility surface*. In a general case, an *implied parameter surface* is obtained. The downside of the root-finding approach is that the process has to be repeated for each contract. Moreover, when the model has multiple parameters, simple line search methods cannot be used.

Alternatively, the model calibration can be formulated as an optimization problem of the form

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i \in [n]} w_i \ell(y_i, f(x_i; \boldsymbol{\theta})), \tag{56}$$

where $\ell : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ is some loss function, and w_i are the contract weights for $i \in [n]$. When the loss function $\ell(y, \hat{y}) = (y - \hat{y})^2$ is selected with uniform weights $w_i = n^{-1}$ for $i = 1, ..., n$, the objective function becomes the familiar mean squared error (MSE). The optimization problem in (56) differs slightly from the root-finding problem in (54): instead of trying to find suitable parameters for each contract and time step separately, the goal is to find one set of parameters that fit multiple observations as well as possible. This calibration can be done over multiple time periods at the same time, or

it can be repeated at each time step, thus producing different $\boldsymbol{\theta}_t^*$ for each t. The problem in (56) can be solved with many optimization algorithms, most of which require the gradient, and sometimes the Hessian, of the objective function with respect to $\boldsymbol{\theta}$.

Another approach, and the approach used in this thesis, is the *inverse map* method in which one tries to solve the optimization problem

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_{i \in [n]} w_i \ell(y_i, f(x_i; g(x_i, y_i; \mathbf{w}))), \tag{57}$$

where $g : \mathbb{R}^{k+1} \to \mathbb{R}^m$ is a function parameterized by the weight vector $\mathbf{w}$. That is, the goal is find a mapping $\boldsymbol{\theta}_i := g(x_i, y_i; \mathbf{w})$ from all possible weights $\mathbf{w}$ such that the pricing function f produces an optimal fit to the market prices. The optimal weights $\mathbf{w}^*$ are typically chosen using some iterative algorithm, such as the *stochastic gradient descent* (SGD) in the case of neural networks (see Section 3.2). It should be noted however, that the problem in (57) is generally non-convex, so a non-global minimum can be obtained from the optimization process. The parameters $\boldsymbol{\theta}_i$ can vary between different contracts and time steps, but only one set of weights, $\mathbf{w}^*$, is needed for the whole dataset. After the model has been fitted, subsequent contracts can be priced by evaluating g and f so that re-calibration is not necessary (assuming that the model generalizes well). The function g is usually restricted to be a member of some function class, such as the class of continuous functions. Figure 9 presents a diagram of the inverse map approach.

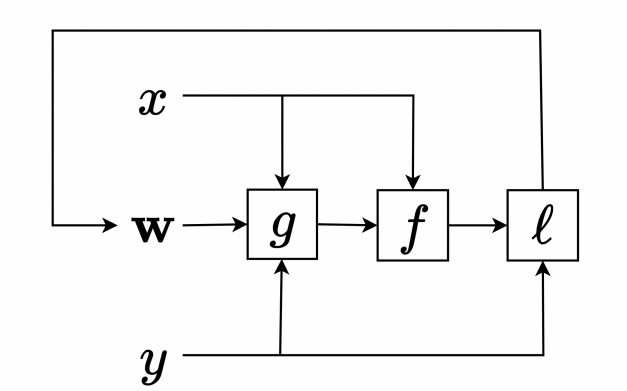

Figure 9: Inverse map diagram. The boxed and unboxed symbols denote functions and variables/parameters, respectively. The market price y, the vector of other variables x (such as in Equation (55)) are fed into the inverse map function g parameterized by the set of weights $\mathbf{w}$. The output of this function is then fed into the pricing formula f along with x, thus producing an estimate $\hat{y} := f(x; g(x, y; \mathbf{w}))$ for the market price y. The goodness of this estimate is assessed by passing y and $\hat{y}$ to the loss function ℓ. Finally, the weights $\mathbf{w}$ can be updated based on the resulting loss.

3.2 Neural Network Approach

Due to their universal approximation property [Hornik et al., 1989] and relative ease of use, *neural networks* (NN) are widely used in many fields, including finance. In recent years, their popularity has only increased due to faster computation, increased memory capacity, machine learning libraries such as TensorFlow, and advancements in research areas such as board games [Silver et al., 2017] and computer vision [Rombach et al., 2022].

Starting from the work of Hutchinson et al. [1994], NNs have also been applied to option pricing. In this area, the bulk of prior research has focused on the forward direction approach [Ruf and Wang, 2020], i.e., the prediction of market prices from the relevant variables. This approach can be useful when the underlying price process is unknown, or when one wants to avoid making too many assumptions about the process. Moreover, domain knowledge can be applied to make the NN models more robust. An example of this is the 'homogeneity hint' proposed by Garcia and Gençay [2000]. A possible drawback of the NN forward direction approach is lack of interpretability, since the effects of different inputs and parameters on the resulting price can be tedious to understand. (Because of this, NNs are often referred to as 'black boxes'.) Additionally, the NN models are not arbitrage-free in general, i.e., they do not satisfy the conditions [Itkin, 2019]

$$\frac{\partial c}{\partial T} \geq 0, \quad \frac{\partial c}{\partial K} \leq 0 \quad \text{and} \quad \frac{\partial^2 c}{\partial K^2} \geq 0 \tag{58}$$

for calls, and

$$\frac{\partial p}{\partial T} \geq 0, \quad \frac{\partial p}{\partial K} \geq 0 \quad \text{and} \quad \frac{\partial^2 p}{\partial K^2} \geq 0 \tag{59}$$

for puts. The conditions correspond to requirements that *calendar spreads*, *vertical spreads* and *butterfly spreads* should have nonnegative prices, respectively, so that free profit cannot be made by holding these positions. The above conditions can be taken into account by introducing soft constraints [Itkin, 2019], or by restricting the NN function class [Dugas et al., 2009]. Like in any constrained optimization however, the in-sample fit may become worse as a consequence.

In this thesis, we are mainly interested in the backward direction approach, since the goal is to recover the parameters of different pricing models. However, when using the inverse map approach given in Equation (57), we also recover the price predictions from the pricing function f. It is important that these predictions are sufficiently accurate, for the obtained parameters would not be reliable otherwise. If the model parameters are treated as constants for each contract and time step, the conditions in (58) and (59) are guaranteed to be satisfied, since the pricing models used (Black-Scholes,Corrado-Su,Heston and

Bates) are arbitrage-free. Next, we outline the inverse map approach using neural networks.

In this thesis, we follow the framework proposed by Andreou et al. [2010] and model the inverse map g as a *multilayer perceptron* (dense feedforward neural network, abbreviated as MLP)

$$g(X; \mathbf{W}) = h_l(X_l W_l + B_l), \tag{60}$$

where $X \in \mathbb{R}^{n \times (k+1)}$ is a dataset of n contracts with $k+1$ variables each (market price y_i plus the other variables x_i), l is the number of non-input layers in the network ($l-1$ is the number of hidden layers), and $\mathbf{W} = \{W_1, ..., W_l\}$ is the set of network weight matrices, where $W_i \in \mathbb{R}^{k_{i-1} \times k_i}$ for $i \in [l]$ with $k_0 = k+1$ and $k_l = m$. Moreover, X_j is defined recursively as

$$X_j = h_{j-1}(X_{j-1} W_{j-1} + B_{j-1}),$$

with the base case $X_1 = X$. Finally, for all $i \in [l]$, h_i and B_i are the activation function and the bias matrix for the i:th layer, respectively, with

$$B_i = \left[b_i^T, \ ... \ , b_i^T \right]^T \in \mathbb{R}^{n \times k_i}$$

consisting of the bias vector $b_i \in \mathbb{R}^{k_i}$ broadcasted across the first dimension (batch dimension). Here, we exploit the fact that g is able to output values for multiple contracts simultaneously (hence the input matrix X instead of the individual rows x_i, and the bias matrices B_i instead of the vectors b_i). When computing the loss function ℓ, a weighted sum of single losses is taken across the batch dimension. It is assumed that both the loss function and the pricing function f are able to handle batch data, which is often automatically achieved when using machine learning library functions.

Since the inverse map g in Equation (60) outputs parameters of a given pricing model, and since reasonable ranges for these parameters are typically known in advance, the final activation functions h_l are restricted to be scaled sigmoid functions given by Equation (22). A suitable range $[a_i, b_i]$ depends on the parameter $\boldsymbol{\theta}_i$. Here we have selected ranges that seem to give good empirical results for the SPX options. The parameter ranges are listed in Table 2. It should be noted however, that different ranges may be required when using other underlying assets. In the case of the Corrado-Su kurtosis μ_4, the range is determined by the envelope in Figure 5. For the Corrado-Su skewness μ_3, Table 2 gives the range *before* applying the mapping in Equation (21). The final range after the mapping varies approximately between -0.9 and 0, depending on the value of the kurtosis. Futhermore, we make the assumption of nonpositive skewness for the underlying log return distribution, so μ_3, the Heston correlation ρ and the Bates jump mean μ_J are restricted to

be nonpositive. Finally, both the Black-Scholes/Corrado-Su volatility σ and the Heston/Bates initial volatility $\sqrt{V_0}$ are restricted to the range $[0.05, 0.8]$.

Table 2: Final layer sigmoid ranges for different parameters.

Model	Parameter	Lower Bound	Upper Bound
Black-Scholes/Corrado-Su	σ	0.05	0.8
Corrado-Su	μ_3	-2.5*	-0.5*
Corrado-Su	μ_4	3	7
Heston/Bates	κ	1	5
Heston/Bates	θ	0.01	0.09
Heston/Bates	V_0	0.0025	0.64
Heston/Bates	ω	0	1
Heston/Bates	ρ	-0.95	0
Bates	λ_J	0	10
Bates	μ_J	-0.02	0
Bates	σ_J	0	0.02

*before applying the mapping in (21)

In the neural network setting, problem (57) can be reformulated as

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} w^T \ell(y, f(X; g(Z; \mathbf{W}))), \tag{61}$$

where g is given by Equation (60), $w \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ are vectors of observation weights and market prices, respectively, and $Z \in \mathbb{R}^{n \times (k_z+1)}$ is a matrix containing variables derived from y and X. The weights $\mathbf{W}$ are obtained using an iterative optimization method, such as the stochastic gradient descent. It is important to note that Z need not contain all variables present in X, and vice versa. The construction of Z is a matter of preference: the variables can be selected based on trial-and-error experiments, or some prior knowledge. In this thesis, X contains the variables in (55), and Z contains the variables

$$z = (F_M, T, \bar{y}, b)\,, \tag{62}$$

where

$$F_M = \frac{S_0 e^{(r-q)T} - K}{S_0 e^{(r-q)T}}$$

is the *forward moneyness* for calls, $\bar{y} = y/S_0$ is the market price normalized by the underlying price, and b is the put-call binary flag (1 for puts, 0 for calls), as before. The put-call flag is used only if the models are fitted using calls and puts simultaneously, otherwise we set $z = (F_M, T, \bar{y})$. A diagram of the 'full network' (the map from X to Z plus the map from Z to $\boldsymbol{\theta} = g(Z; \mathbf{W})$) is

given in Figure 10.

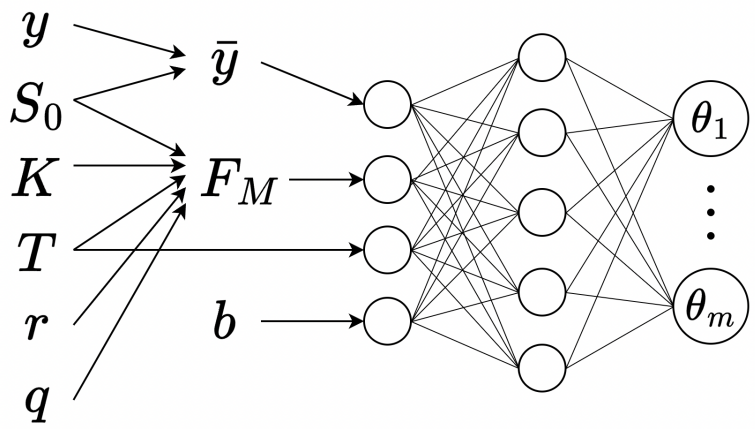

Figure 10: Diagram of the full network in the case of a single hidden layer and five hidden units. First, the vector of variables $x = (S_0, K, T, r, q)$ and the market price y are converted to the preprocessed variables $z = (\bar{y}, F_M, T, b)$. Then, these variables are passed as inputs to the MLP (dense network) that outputs the parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_m)$ of a given pricing model. (Technically, the preprocessing step can be considered as part the inverse map g, but here g contains only the MLP part.)

3.3 Model Training

There are a lot of choices to be made when implementing and training the inverse map g. First, one should choose the preprocessed variables z and suitable parameter ranges for the final sigmoid activation functions, which we have already done. There is also the choice of the activation functions for the hidden layers. Additionally, the loss function and the optimization method should be decided for the training process. Finally, there are several hyperparameters associated with the network structure and the training process. For the network, the number of layers and the number of units for each layer must be selected. For the training process, one needs to specify the number of *epochs* (passes trough the entire training set), *batch size* (number of observations for which the stochastic gradient is computed at each time), sample weights w in Equation (61), and the *learning rate* for the optimization algorithm.

First, for the network structure, the hidden layer activation functions are selected to be hyperbolic tangent functions (tanh, ranged between -1 and 1). The Leaky ReLU (Leaky Rectified Linear Unit) activation is also tried, but this results in worse model performance. Moreover, the number of units in hidden layers are restricted to be the same, i.e., $k_1 = ... = k_{l-1} = k$ for some constant k. Now, for each model, we try different values for the number of

non-input layers l and the number of units k, and select the values which give best out-of-sample results. More specifically, we select the pair $(l, k) \in S_l \times S_k$ from all combinations $l \in S_l$ and $k \in S_k$, such that some metric is minimized or maximized over multiple validation splits. Here, we set $S_l = \{2, 3\}$ (1 or 2 hidden layers) and $S_k = \{6, 8, 10\}$. Thus, the number of total weights in the network ranges between 31 and 248, depending on the model and whether the put-call flag is used (see Table 3). The network is restricted to be relatively small, because this results in faster convergence during training, while still obtaining a good fit to the market prices. This is of great interest to us, since the training process is run for multiple models, option types (only calls, only puts, and both puts and calls), time splits and initial network weights. Finally, the problem of overfitting does not seem to be severe for the number of weights used here. Hence, regularization techniques, such as weight decay and dropout, are not employed in this thesis.

Table 3: Minimum and maximum number of network weights for each model, when the network inputs in (62) are used. The values in parentheses correspond to the case where the put-call flag is dropped.

Model	Min. #weights	Max. #weights
Black-Scholes	37 (31)	171 (161)
Corrado-Su	51 (45)	193 (183)
Heston	65 (59)	215 (205)
Bates	86 (80)	248 (238)

For the loss function in Equation (61), we choose the quadratic loss $\ell(y, \hat{y}) = (y - \hat{y})^2$, and for the sample weights, we select

$$w_i = \frac{1}{n\sqrt{y_i}},$$

where n is again the number of contracts (in a single batch), and y_i is the market price of the i:th observation. Thus, for a market price vector $y \in \mathbb{R}^n$ and a price estimate vector $\hat{y} := f(X; g(Z; \mathbf{W}))$, the total loss L is given by

$$L = \frac{1}{n} \sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{\sqrt{y_i}}, \tag{63}$$

which is the mean squared error (MSE) weighted by the inverse square root of prices $y_i^{-1/2}$. This weighting (partly) compensates for the fact that the absolute difference $|y_i - \hat{y}_i|$ tends to increase as y_i increases, which leads to less expensive options contributing less, and more expensive options contributing more, to the total MSE loss. In other words, the MSE is 'biased' towards

ITM and long maturity options which are generally pricier than OTM and short maturity options. Since the loss is quadratic, the weights $y_i^{-1/2}$ do not eliminate this bias. To achieve that, squared inverse prices y_i^{-2} could be used as weights instead. Alternatively, we could replace the weighted MSE by the mean absolute percentage error (MAPE)

$$\text{MAPE} = 100\% \cdot \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|. \tag{64}$$

However, both alternatives seem to result in worse overall model performance. Moreover, the inverse price weights y_i^{-1} seem to result in more erratic loss trajectories during training, so those weights are not used either. Thus, we settle for the loss in Equation (63).

For optimization, the popular *Adam* algorithm, presented by Kingma and Ba [2014], is used. When updating the network weights $\mathbf{W}$, the Adam optimizer approximates both the first and the second moments of the gradients, which can lead to faster and better convergence during training. Indeed, we find that the Adam optimizer is superior to the SGD optimizer for solving the problem in Equation (61). Furthermore, by periodically reducing the learning rate of the optimizer, λ_L, a better fit to the data is obtained. When a constant learning rate is used during the training process, the loss seems to get stuck at a higher value than what would be achieved if the learning rate was lowered. Additionally, if too large of a learning rate is used at the beginning of training, the Heston and the Bates models can suffer from the *exploding gradient problem*. Hence, a short warm-up period is employed at the start of each training process. That is, we start with a relatively small learning rate, linearly increase it during the first few epochs, and then gradually reduce the rate during the remaining epochs. For all (4) models, we start with the rate $\lambda_L = 0.02$, linearly increase it to $\lambda_L = 0.2$ during the first 20 epochs (increase by 0.02 after every other epoch), and then reduce the value by 20% after every 20th epoch. In total, 200 epochs are run for each training process (one model and one training split). For each batch, we sample 32 different time steps, and for each time step, include 42 contracts with predetermined maturities and strikes (see Section 4.1). Thus, the effective batch size is 1134. The number of time steps per training set is at least 375 (see Section 4.2), so the number of batches per epoch is at least 11.

It should be noted that since the training is partly stochastic due to the random batches and initial network weights, different local minima of Equation (61) can be found by repeating the training process. Therefore, for each model, training split and set of hyperparameters (l and k), the model is trained multiple times using different batches and initial weights. Here, the non-bias weights are initialized using the Xavier (uniform) initialization [Glorot and Bengio, 2010], and the bias weights are initialized to zero. The randomness

is controlled by setting a different *seed* at the start of each run. This allows us to investigate the sensitivity of the results to the random training process, while also being able to reprocude the results.

Finally, we find that network input normalization is important for achieving convergence during training. Because of this, the variables in (62), with the exception of the put-call flag, are converted to their z-scores by

$$\hat{z}_i = \frac{z_i - \bar{z}_i}{s_i},$$

where $\bar{z}_i$ and s_i are the sample mean and standard deviation of the i:th variable z_i, respectively. For a single train-validation-test split, $\bar{z}_i$ and s_i are calculated from only the training set, after which the z-scores are computed for the training, validation and test sets using these fixed values. Table 4 summarizes the choices made in this chapter.

Table 4: Summary of the choices regarding the network structure and the training process.

Category	Factor	Choice
Network	type	multilayer perceptron
	inputs	$F_M, T, \bar{y}, (b)$
	input normalization	z-score
	final activation	scaled sigmoid
	hidden activations	tanh
	hidden layers	1 or 2
	units per hidden layer	6,8 or 10
Training	loss	MSE weighted by the inverse square root of market prices
	optimizer	Adam
	initial learning rate	0.02
	learning rate schedule	increase by 0.02 every other epoch until epoch 20, reduce by 20% every 20th epoch after that
	weight initialization	Xavier (non-bias weights), zero (bias weights)
	regularization	-
	epochs	200
	batch size	1134 (32 time steps w/ 42 contracts)
	batches per epoch	≥ 11

3.4 Implementation Considerations

The above framework is quite flexible, and suitable for many option pricing models. A benefit of the framework is that most parts of the implementation, such as the code for model training and testing, remain the same between different pricing models. In fact, when introducing a new pricing model, only the pricing function and suitable parameter ranges need to defined.

A key requirement for a given pricing function f is that the network weights $\mathbf{W}$ can be optimized by minimizing the loss given by the loss function ℓ. In the neural network context, this practically means that the gradient of f (and ℓ) with respect to $\mathbf{W}$ can be computed. The TensorFlow (TF) library and its extension TensorFlow Probability (TFP) provide the necessary functions, and the gradients of these functions, for implementing the option pricing formulas from Chapter 2. In this thesis, we use version 2.10.0 of TF and version 0.18.0 of TFP. The models are implemented using the programming languague R, and the TF/TFP functions are accessed using the `reticulate` package.

Assume that the TF and TFP packages have been imported in Python as `tf` and `tfp`, respectively. In the case of the Black-Scholes and Corrado-Su models, the call prices can be computed using basic elementary operations (`tf.add`, `tf.subtract`, `tf.multiply` and `tf.divide`), square root (`tf.sqrt`), exponentiation (`tf.exp`), logarithm (`tf.math.log`), and the normal density and cumulative distribution function (methods `prob` and `cdf` of the TFP class `tfp.distributions.Normal`, respectively). Moreover, the Gram-Charlier adhoc mapping from Equation (21) can be implemented by precomputing the skewness-kurtosis pairs (s_i, k_i) which can then be used to calculate the grid points a_j and b_j (in non-tensor form). Then, the skewness can be scaled using the sigmoid function (`tf.sigmoid`). By Table 2, the kurtosis is already in the valid range.

In the case of the Heston and Bates models, the hyperbolic functions required in Equation (43) are available in TF (`tf.math.sinh` and `tf.math.tanh`). Moreover, TF handles the computations between complex numbers automatically, provided that the real-valued tensors have been converted to complex tensors using the `tf.complex` function. The numerical integrals in Equations (41) and (52) are easily obtained by precomputing the Gauss-Legendre nodes and weights, and then calculating the weighted sum in Equation (40) using `tf.reduce_sum`. We find that 30 nodes and weights are enough for sufficient convergence. Finally, we set the inverse map to output the long-term and initial *volatilities* $\sqrt{\theta}$ and $\sqrt{V_0}$ instead of the variances θ and V_0. This change of variables seems to give more reasonable values for θ and V_0.

The availability of the pricing function f and its gradients is not enough to obtain a satisfactory solution to the problem in (61). Namely, the pricing function and the gradients should be numerically stable, so that the iterative optimization method is able to convergence. For the same reason, it is desirable

that the magnitude (norm) of the gradients is not too big or small. The option pricing models in Chapter 2 have been selected partly because they satisfy these requirements. For the opposite reason, some alternative models are excluded from consideration in this thesis. For instance, the *Borland-Bouchaud model* [Borland and Bouchaud, 2004] produces fairly good results to the problem in (61), but it is a bit too unstable numerically, and requires more supervision than the other models during training. Additionally, we experiment with the *Constant Elasticity of Variance* (CEV) [Cox, 1997] and the *Variance Gamma* (VG) [Madan et al., 1998] models, but find no satisfactory solution using TensorFlow. In particular, the computational challenges are caused by the non-central Chi-squared distribution in the case of the CEV model, and the modified Bessel function of the second kind in the case of the VG model.

4 Numerical Results

4.1 Data

In this thesis, the neural network models are calibrated to the S&P 500 index (SPX) options, issued by the Chicago Board Options Exchange (Cboe). The SPX options are selected, since the market for them is extremely liquid, which results in relatively small bid-ask spreads. Furthermore, the options are European, so their prices can be estimated by the pricing models presented in Chapter 2. Hence, SPX options provide a good testing ground for the NN framework introduced in Section 3.2.

The options data is obtained from the Cboe DataShop [Cboe, 2023], It contains quotes from the 'SPX' options chain between January 2012 and January 2023. The SPX options chain consists of the standard options (root ticker 'SPX') which expire on the 3rd Friday of each month, and the weekly and end-of-month options (root ticker 'SPXW') which can expire on different days of the week. The former options are settled in the morning of the expiration date, the latter at the close of the market in the afternoon (Easter Standard Time). The time frequency of the data is one quote per hour, so there are about 6-7 daily observations (quotes) for each option contract. Each observation consists of the following fields:

- quote timestamp t
- underlying price S_0
- expiration time τ
- strike price K
- bid b
- bid size n_b
- ask $a \geq b$
- ask size n_a

By taking the difference between the expiration date and the timestamp, we obtain the time to maturity $T := \tau - t$ in years. Moreover, when training and testing the models, the option market price y is defined as the mid price of the bid and ask, $y := (b + a)/2$.

In addition to the options data, the option pricing models require estimates for the risk-free rate r and the dividend yield q of the S&P 500 index. In this thesis, the risk-free rate is calculated from the US Treasury Security yields at different constant maturities. These yields are collected from the Federal Reserve Economic Data [FRED, 2023]. We use the 1 month ('DGS1MO'), 3 month ('DGS3MO'), 6 month ('DGS6MO') and 1 year ('DGS1') maturities. For each option contract, r is calculated by interpolating the fixed maturity yields at the contract's maturity. Here, yield extrapolation is not permitted, so option maturities are restricted between 1 and 12 months. For interpolation, a cubic spline is used (see Figure 11). The daily dividend yield is collected

from the GuruFocus website [GuruFocus, 2023]. The values are obtained by averaging the yields of the companies in the the S&P 500 index, weighted by the market capitalization of each company. The dividend yield of each company is the ratio of the last yearly dividend and the current share price. Figure 12 shows the percentage yield between the years 2012 and 2022.

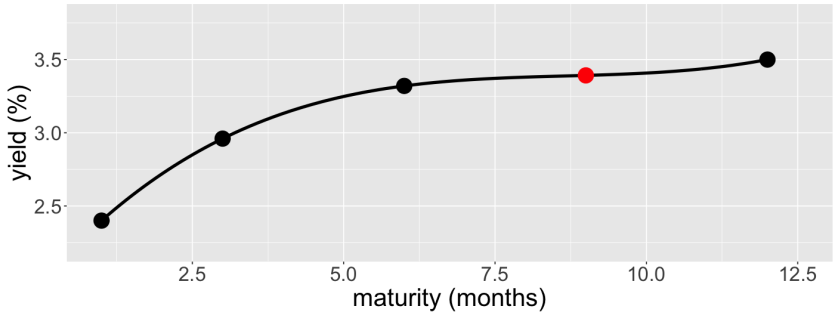

Figure 11: Cubic spline fit to the fixed maturity yields at 1, 3, 6 and 12 months (black points). For a option contract with a maturity of 9 months (red point), the risk-free rate r is estimated by interpolation. Here, we obtain the rate $r \approx 3.39\%$.

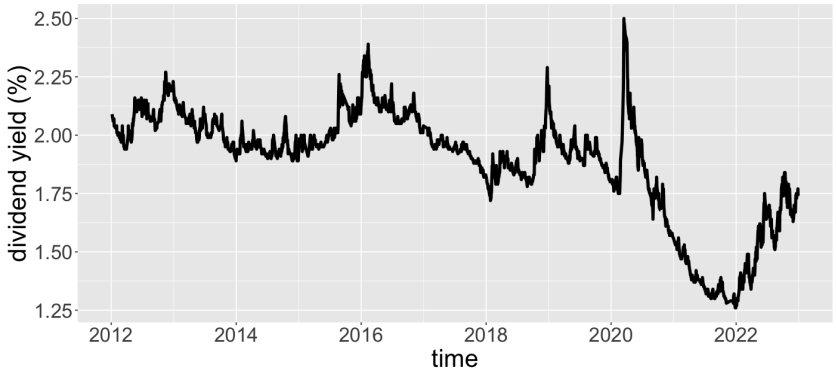

Figure 12: Combined daily dividend yield of the companies in the S&P500 index from 2012 to 2022.

Before the model training process, the option data is filtered using the following rules. First, we remove the observations for which either the bid size n_b or the ask size n_a is zero. Additionally, we only consider the observations that satisfy the arbitrage restrictions

$$y_c \geq \max(0, \mathrm{F_{IV}}),$$
$$y_p \geq \max(0, -\mathrm{F_{IV}}),$$

where

$$\mathrm{F_{IV}} = S_0 e^{-qT} - K e^{-rT}$$

is the 'forward intrinsic value' of the call option, and y_c and y_p are the market (mid) prices for the call and the put, respectively. Since the payoff from an option is always nonnegative, the restrictions $y_c \geq 0$ and $y_p \geq 0$ are obvious. When $\mathrm{F_{IV}} > 0$ ($\mathrm{F_{IV}} < 0$), the restriction $y_c \geq \mathrm{F_{IV}}$ ($y_p \geq -\mathrm{F_{IV}}$) follows from the put-call parity in Equation (1). Since the option pricing models from Chapter 2 assume frictionless markets (markets with unlimited funding, lending and shorting, and no transaction costs), they are not well-equipped to handle options that do not satisfy these restrictions. Acknowledging these limitations, we remove such options from the final dataset.

After the above rules, the data is further filtered by forming a $N_T \times N_K$ grid of N_T different maturities and N_K different strikes for each time step t. The same grid is constructed for calls and puts separately, so each time step contains $2 \cdot N_T \cdot N_K$ observations. This is done for couple of reasons. First, it ensures that equal number of options from different categories (puts/calls, short/long maturities, ITM/ATM/OTM strikes) are used during the training and testing of models. Second, it enables the construction of discrete implied parameter surfaces for each time step. To normalize the grid, we work with the option *moneyness*

$$M = d_M \cdot \frac{S_0 - K}{S_0}$$

instead of the strike K, where $d_M = 1$ for calls and $d_M = -1$ for puts. Let G be the target grid of maturity-moneyness pairs. For each t and pair $(\bar{T}, \bar{M}) \in G$, we first find the set of contracts with time to maturity T such that the distance $T - \bar{T}$ is minimized subject to $\bar{T} \leq T \leq \bar{T} + \delta(\bar{T})$, where $\delta(\bar{T})$ is a threshold function. Then, from this set of contracts, we select the one with moneyness M such that the absolute difference $\left|M - \bar{M}\right|$ is minimized. If no contract is found for all pairs $(\bar{\tau}, \bar{M}) \in G$ using these rules, the time step t is discarded entirely, because we are only interested in complete grids. However, the necessary contracts are found for the majority of all time steps. Finally, to reduce the number of observations for faster training, we only include the last time step of each trading day.

In this thesis, we use a 6×7 grid visualized in Figure 13. The vector of target maturities (in days)

$$G_T = (30, 60, 90, 120, 180, 270)$$

has been selected to obtain sufficient number of complete grids for different time steps. As can be seen from Figure 13, the moneyness range is widened as the target maturity $\bar{T} \in G_T$ increases. For each index $j \in [6]$ of $\bar{T}$ in G_T, the vector of target moneyness levels is given by

$$G_K(j) = k_j \, (-3, -2, -1, 0, 1, 2, 3)$$

for a constant vector

$$k = 10^{-2} \cdot (1.33, 1.83, 2.33, 2.67, 3.33, 4.00)$$

The relationship between k_j and $\bar{T}$ is approximately $k_j \sim \sqrt{\bar{T}}$, so k_j scales linearly with the total asset volatility $\sigma\sqrt{t}$. Moreover, k is scaled so that the furthest ITM and ATM options have moneyness of approximately $\pm 4\%$ for the shortest maturities, and $\pm 12\%$ for the longest maturities. In order, the (6) maturity levels are labeled `t1`,`t2`,`t3`,`t4`,`t5` and `t6`, and the (7) moneyness categories are labeled `otm3`,`otm2`,`otm1`,`atm`,`itm1`,`itm2` and `itm3`.

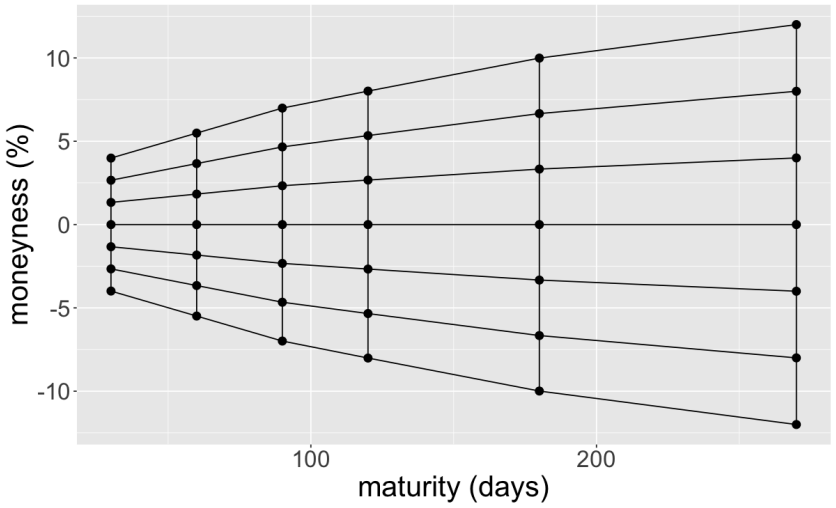

Figure 13: The maturity-moneyness grid.

Finally, for the maturity threshold function, we use

$$\begin{aligned} \delta(30) &= 20, & \delta(60) &= 20, \\ \delta(90) &= 20, & \delta(120) &= 50, \\ \delta(180) &= 80, & \delta(270) &= 90, \end{aligned}$$

to ensure small gaps (10 days) between the maturity levels, but also to ensure that the grid is not too sparse in the (T, M)-space. Additionally, the last threshold is used to obtain an upper bound of approximately one year. Figure 14 shows the data points that have been obtained using the grid described above. The final dataset contains 2015 time steps with $2 \cdot 6 \cdot 7 = 84$ data points each, so the number of total data points is 169260, half of which are calls/puts.

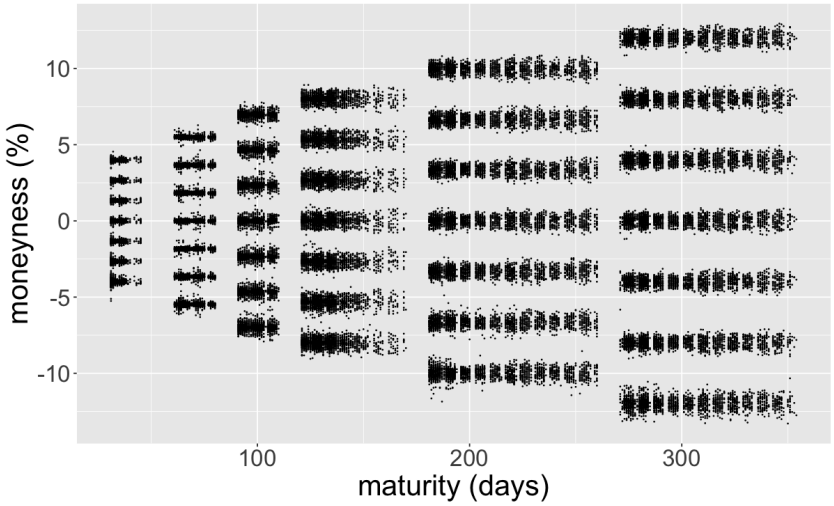

Figure 14: The final data points in the (T, M)-space.

4.2 Model Evaluation

In this section, we evaluate the performance of the Black-Scholes (B-S), Corrado-Su (C-S), Heston and Bates models, when applied to the neural network inverse map framework presented in Section 3.2. For each model, the evaluation is carried out using (i) only calls, (ii) only puts, and (iii) both calls and puts. When evaluating the pricing accuracy of the models, the following metrics are used: the (i) Root Mean Squared Error RMSE $= \sqrt{\text{MSE}}$, (ii) the MAPE from Equation (64), (iii) the absolute error relative to the bid-ask spread

$$\text{errSpread} := \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{a_i - b_i}, \tag{65}$$

where b_i and a_i are the bid and ask prices of observation i, respectively, and (iv) the percentage of predictions inside the spread

$$\text{pSpread} := 100\% \cdot \frac{1}{n} \sum_{i=1}^{n} I(b_i \leq \hat{y}_i \leq a_i),$$

where I is the indicator function. The metrics above are selected for the following reasons. First, the RMSE is closely related to the weighted MSE loss that is used during training. We use the RSME instead of the MSE, because the values of the former are in dollar scale. Moreover, since the RMSE considers the squared pricing errors, it is sensitive to large absolute errors. However, it is also biased towards expensive options, as mentioned in Section 3.3. Thus, to better examine the relative pricing errors, we also include the MAPE in our study. The downside of the MAPE is that it does not take

into account the bid-ask spreads of the options. For this, the errSpread and pSpread metrics are included. Here, we consider the price predictions inside the spread to be 'ideal', in the sense that even market participants cannot agree on a fair price between the bid and ask prices. Hence, the pSpread metric gives the percentage of 'ideal' predictions. An errSpread score of 0.5 is considered to be excellent, because this is the value that would be achieved if the model gave predictions identical to the bid or ask prices in the market.

To assess the performance of the models in different market conditions, the models are tested on multiple time intervals. More specifically, we use 10 train-validation-test splits, listed in Table 5. The first split starts from July 2015, and for each subsequent split, the time window is slid forward 6 months. Each 2-year long train interval is followed by a 5-month long validation interval, followed by a 6-month long test interval. The validation and test intervals are non-overlapping, and the last test interval ends at the end of 2022. Additionally, a 14 day gap is left between each train and validation interval, and each validation and test interval. This is done to mitigate possible underestimation of the out-of-sample error due to autocorrelations between consecutive observations. The validation intervals are used to select the best version of each model, i.e., the best pair of hyperparameters (l, k) (number of non-input layers, and number of units for each hidden layer, respectively). This is done by averaging some metric over all validation intervals and seeds that control the random training process, and then selecting the hyperparameters corresponding to the best value. Here, we choose (l, k) which minimize the errSpread metric from Equation (65). For each model, training interval and pair of hyperparameters, the training process is repeated using 5 different seeds. Finally, the best version of each model is evaluated on the test intervals.

Table 5: Train-validation-test splits in 'DD/MM/YY' format.

	Train	Valid.	Test
1	01/07/15 - 30/06/17	15/07/17 - 17/12/17	01/01/18 - 30/06/18
2	01/01/16 - 31/12/17	15/01/18 - 16/06/18	01/07/18 - 31/12/18
3	01/07/16 - 30/06/18	15/07/18 - 17/12/18	01/01/19 - 30/06/19
4	01/01/17 - 31/12/18	15/01/19 - 16/06/19	01/07/19 - 31/12/19
5	01/07/17 - 30/06/19	15/07/19 - 17/12/19	01/01/20 - 30/06/20
6	01/01/18 - 31/12/19	15/01/20 - 16/06/20	01/07/20 - 31/12/20
7	01/07/18 - 30/06/20	15/07/20 - 17/12/20	01/01/21 - 30/06/21
8	01/01/19 - 31/12/20	15/01/21 - 16/06/21	01/07/21 - 31/12/21
9	01/07/19 - 30/06/21	15/07/21 - 17/12/21	01/01/22 - 30/06/22
10	01/01/20 - 31/12/21	15/01/22 - 16/06/22	01/07/22 - 31/12/22

At minimum, the train, validation and test intervals contain 375,64 and 78 complete time steps (time steps for which a complete grid of contracts can be constructed, see Section 4.1), respectively. Since each time step contains $6 \times 7 = 42$ contracts, the minimum number of data points is 15750 for train

intervals, 2688 for validation intervals, and 3276 for test intervals. Table 6 summarises the number of observations. Moreover, Figure 15 shows the price path of the S&P 500 index during the test intervals.

Table 6: Minimum, maximum and average number of observations, and the total number of unique data points, across all time splits.

	Train	Valid.	Test
Min.	15750	2688	3276
Avg.	17686	3713	4423
Max.	18774	4326	5124
Total	56322	37128	44226

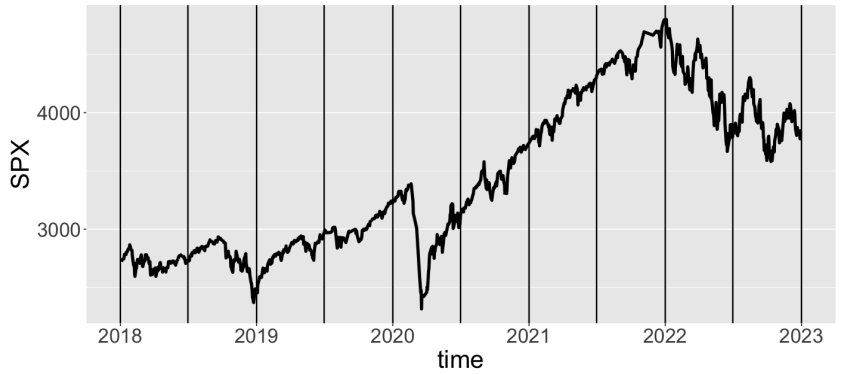

Figure 15: S&P500 index (SPX) prices during test intervals. The non-overlapping intervals are separated by vertical lines.

Next, we summarise the results that have been obtained by fitting the pricing models to *calls only*. This is done for a couple of reasons. First, we omit the examination of the puts only case, because this section would be too lengthy otherwise, and more importantly, because the overall results are quite similar to the ones presented here. However, a notable exception is the case of OTM contracts, where a smaller percentage error is obtained for puts than for calls. This is because OTM calls tend to be cheaper than OTM puts (implying a negatively skewed log return distribution), which makes the pricing of OTM calls more challenging using the loss in Equation 63. Overall, the results are slightly better for puts than they are for calls. Appendix A lists selected evaluation results for puts.

Second, we find that better results are obtained when the models are fitted to calls and puts separately. In theory, the models should be able to handle both contract types simultaneously due to the put-call parity, but in practice, this results in worse performance both in-sample and out-of-sample. It is possible that the choices regarding the network structure and the training process (that were made in Chapter 3) are partly responsible for this, and that a different conclusion would be reached with alternative choices, such as

a different set of network inputs. Nevertheless, here we also omit the results corresponding to both calls and puts.

Table 7 shows the errSpread values for all pairs of (l, k), averaged over all validation splits and (5) seeds. The bottom row gives the average value for each pair (over all models), and the rightmost column shows the average value for each model (over all pairs). From the table, it is clear that larger networks produce superior results compared to the smaller networks. In the case of all four models, the hyperparameter pair $(3, 10)$ gives the smallest validation error. Moreover, it seems that a single hidden layer ($l = 2$) is unable to fully capture the behaviour of the inverse pricing formula across multiple maturities and strikes, since the error is considerable larger than in the case of two hidden layers ($l = 3$). From the table, it can also be seen that the Black-Scholes model has the largest, and the Bates model has the smallest, average validation error. However, the Heston model has the smallest validation error when only the best hyperparameters are considered.

Table 7: Relative spread errors (errSpread) for different pairs of hyperparameters l (number of non-input layers) and k (units per hidden layer), averaged over all validation splits and random seeds. The errors are colored from red (largest) to green (smallest).

Model	l=2 k=6	l=2 k=8	l=2 k=10	l=3 k=6	l=3 k=8	l=3 k=10	Avg.
Black-Scholes	3.318	2.912	2.242	1.303	1.042	0.956	1.962
Corrado-Su	2.394	1.999	2.131	1.418	0.958	0.863	1.627
Heston	2.025	1.834	1.617	1.172	1.002	0.725	1.396
Bates	1.878	1.459	1.274	1.052	0.930	0.802	1.233
Avg.	2.404	2.051	1.816	1.236	0.983	0.837	1.554

From now on, we only consider the best hyperparameter version (that is, the case $l = 3$, $k = 10$) of each model. Additionally, we run each version with 7 more random seeds, so the total number of seeds per model and split is 12. Table 8 shows the model metrics averaged over the train, validation and test intervals, and the random seeds. For the train intervals, the order from the best to the worst model is the same in all four categories (metrics): (1) Bates, (2) Heston, (3) Corrado-Su, (4) Black-Scholes. Thus, more complex models produce better train metrics across the board. In the validation and test case however, the order is not so clear. For the validation intervals, the Bates model is the best in three categories (MAPE, errSpread and pSpread), and the C-S model has the best RMSE. On the other hand, the Bates model has the worst validation RMSE, while the B-S model has the weakest validation results in the remaining categories. For the test intervals, the Bates and C-S models are the best in two categories each (MAPE and errSpread, and RMSE and errSpread, respectively). Conversely, the B-S model gives the least accurate

results in three categories (MAPE, errSpread and pSpread), and the Heston model has the worst test RMSE. Overall, the B-S model seems to have the weakest results. The remaining models are a bit more evenly matched, with the Bates and C-S models having a slight edge over the Heston model.

Overall, the metrics are worse for the validation and test intervals than they are for the train intervals, with the exception of the MAPE. Hence, the models seem to slightly overfit to the training sets. With that said, the overfitting is not too severe, because all the models give fairly good out-of-sample predictions. For instance, the out-of-sample predictions land inside the spread over 50% of the time in the case of all models, and both validation and test intervals. Additionally, the average out-of-sample relative spread error satisfies errSpread < 1, with the exception of the B-S test value. This means that the average predictions are not too far from the actual bid or ask prices, although the errors are larger than the 'ideal' errSpread score of 0.5. On average, the test results are slightly worse than the validation results, which suggests a slight performance decay as time passes. This indicates that a single calibration does not suffice indefinitely, and that the models should be re-calibrated as enough time has passed from the training set. However, a suitable calibration frequency is likely less than that of the constant parameter setting (problem in (56)), where the models are often re-calibrated daily.

Table 8: Average model metrics over train, validation and test sets. For each metric, the model giving the best test result is colored green, and the model corresponding to the worst test result is colored red.

Model	RMSE	MAPE	errSpread	pSpread
Black-Scholes	train : 0.80 valid : 2.35 test : 2.79	train : 2.61 valid : 2.81 test : 1.74	train : 0.60 valid : 0.93 test : 1.01	train : 65.51 valid : 53.95 test : 50.24
Corrado-Su	train : 0.70 valid : 2.11 test : 2.07	train : 2.37 valid : 2.50 test : 1.45	train : 0.52 valid : 0.81 test : 0.83	train : 70.29 valid : 57.99 test : 55.11
Heston	train : 0.66 valid : 2.86 test : 3.17	train : 1.74 valid : 2.10 test : 1.42	train : 0.43 valid : 0.81 test : 0.88	train : 74.54 valid : 62.05 test : 58.51
Bates	train : 0.63 valid : 2.90 test : 3.08	train : 1.58 valid : 1.96 test : 1.34	train : 0.40 valid : 0.78 test : 0.85	train : 76.50 valid : 63.86 test : 59.70

Table 9 shows the average model metrics for different test years. From the rightmost column, it is clear that the average model performance does not stay the same from year to year. In particular, the performance is by far the worst during the year 2020, where the Heston and Bates models suffer the largest drop in performance. This is not very surprising, since 2020 was a very

abnormal and volatile market year due to the start of the COVID-19 pandemic (see Figure 15). In other words, the models seem to perform the worst during the most extreme market conditions, at least in this particular test period. In addition, more complex models are more prone to overfitting due to the larger number of model parameters, which can explain the underwhelming results of the Bates and Heston models during 2020. This performance drop-off is of course undesirable, but also to be expected, since the preceding training data does not contain such extreme market conditions. In any case, it is important to acknowledge the limitations, and not to blindly trust the outputs, of the models. On the other hand, the best test results are achieved for multiple metrics during the other bad market years, namely 2018 and 2022. Hence, it cannot be concluded that the models do not work in all market downtrends. Table 9 also shows that 2020 has a significant negative impact on the average test metrics in Table 8, and that the average test performance is noticeably better during the more 'normal' market years. For instance, when 2020 is excluded, the average test errSpread of the Bates model is an excellent 0.49.

Table 9: Average test metrics for different years. For each test period (row) and metric, the models are ranked using numbers from 1 to 4. Then for each row, the model with the smallest average rank is colored green, and the model with the largest average rank is colored red. In the case of a tie, both models are colored. For brevity, the results are presented annually, even though the test intervals are 6 months long.

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
2018	RMSE : 0.80 MAPE : 1.77 errSpread : 0.52 pSpread : 66.12	RMSE : 0.69 MAPE : 1.57 errSpread : 0.45 pSpread : 70.67	RMSE : 0.65 MAPE : 1.28 errSpread : 0.38 pSpread : 76.11	RMSE : 0.63 MAPE : 1.19 errSpread : 0.36 pSpread : 77.72	RMSE : 0.69 MAPE : 1.45 errSpread : 0.43 pSpread : 72.66
2019	RMSE : 0.75 MAPE : 1.90 errSpread : 0.81 pSpread : 54.14	RMSE : 0.71 MAPE : 1.72 errSpread : 0.74 pSpread : 56.60	RMSE : 0.60 MAPE : 1.36 errSpread : 0.60 pSpread : 62.28	RMSE : 0.55 MAPE : 1.18 errSpread : 0.54 pSpread : 65.62	RMSE : 0.65 MAPE : 1.54 errSpread : 0.67 pSpread : 59.66
2020	RMSE : 9.87 MAPE : 3.03 errSpread : 2.22 pSpread : 24.50	RMSE : 6.89 MAPE : 2.29 errSpread : 1.72 pSpread : 28.51	RMSE : 12.69 MAPE : 3.09 errSpread : 2.32 pSpread : 28.25	RMSE : 12.34 MAPE : 2.99 errSpread : 2.27 pSpread : 28.52	RMSE : 10.45 MAPE : 2.85 errSpread : 2.13 pSpread : 27.44
2021	RMSE : 1.42 MAPE : 1.49 errSpread : 0.99 pSpread : 42.19	RMSE : 1.22 MAPE : 1.26 errSpread : 0.82 pSpread : 47.28	RMSE : 1.16 MAPE : 1.03 errSpread : 0.71 pSpread : 50.30	RMSE : 1.12 MAPE : 1.01 errSpread : 0.70 pSpread : 50.55	RMSE : 1.23 MAPE : 1.20 errSpread : 0.80 pSpread : 47.58
2022	RMSE : 1.12 MAPE : 0.51 errSpread : 0.53 pSpread : 64.26	RMSE : 0.85 MAPE : 0.41 errSpread : 0.41 pSpread : 72.49	RMSE : 0.74 MAPE : 0.35 errSpread : 0.37 pSpread : 75.62	RMSE : 0.76 MAPE : 0.35 errSpread : 0.37 pSpread : 76.09	RMSE : 0.87 MAPE : 0.40 errSpread : 0.42 pSpread : 72.11
Avg.	RMSE : 2.79 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 2.07 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.17 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 3.08 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.78 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89

Tables 10 and 11 present the test metrics for different levels of maturity and moneyness, respectively. In both tables, the levels (rows) are labeled using the labeling from Section 4.1. From the tables, it can be seen that the relative errors (MAPE and errSpread) tend to decrease, and the absolute error (RMSE) tends to increase, with maturity and moneyness. The latter observation is not surprising, because the prices of options tend to also increase with maturity and moneyness. The increasing relative errors can be attributed to the biased loss in Equation (63), but also to the fact that the pricing of ITM options is often easier than the pricing of OTM options. An accurate pricing of ITM options in terms of percentage error is relatively easy, because as the moneyness of the option increases, the price of the option approaches its intrinsic value (the probability of the option expiring worthless becomes negligibly small). Conversely, the value of a deep OTM option can be close to zero, which makes it difficult to produce price predictions with small percentage error.

Table 10: Average test metrics for different maturity levels. Here, we use the same ranking system for coloring as in Table 9.

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
t1	RMSE : 1.88 MAPE : 3.56 errSpread : 1.52 pSpread : 30.50	RMSE : 1.85 MAPE : 3.14 errSpread : 1.29 pSpread : 35.86	RMSE : 2.53 MAPE : 2.71 errSpread : 1.23 pSpread : 39.08	RMSE : 2.39 MAPE : 2.49 errSpread : 1.15 pSpread : 40.95	RMSE : 2.16 MAPE : 2.98 errSpread : 1.30 pSpread : 36.60
t2	RMSE : 1.71 MAPE : 1.55 errSpread : 1.03 pSpread : 45.86	RMSE : 1.44 MAPE : 1.34 errSpread : 0.84 pSpread : 51.43	RMSE : 2.53 MAPE : 1.45 errSpread : 0.99 pSpread : 51.64	RMSE : 2.50 MAPE : 1.38 errSpread : 1.00 pSpread : 53.29	RMSE : 2.04 MAPE : 1.43 errSpread : 0.97 pSpread : 50.55
t3	RMSE : 1.94 MAPE : 1.37 errSpread : 0.94 pSpread : 53.18	RMSE : 1.46 MAPE : 1.15 errSpread : 0.76 pSpread : 57.18	RMSE : 2.55 MAPE : 1.14 errSpread : 0.84 pSpread : 61.23	RMSE : 2.48 MAPE : 1.06 errSpread : 0.81 pSpread : 61.93	RMSE : 2.11 MAPE : 1.18 errSpread : 0.84 pSpread : 58.38
t4	RMSE : 2.28 MAPE : 1.28 errSpread : 0.99 pSpread : 53.13	RMSE : 1.64 MAPE : 1.02 errSpread : 0.79 pSpread : 57.57	RMSE : 2.77 MAPE : 1.05 errSpread : 0.87 pSpread : 62.44	RMSE : 2.66 MAPE : 0.97 errSpread : 0.81 pSpread : 63.30	RMSE : 2.34 MAPE : 1.08 errSpread : 0.87 pSpread : 59.11
t5	RMSE : 3.02 MAPE : 1.24 errSpread : 0.96 pSpread : 58.10	RMSE : 2.14 MAPE : 0.99 errSpread : 0.76 pSpread : 61.95	RMSE : 3.32 MAPE : 1.04 errSpread : 0.83 pSpread : 66.23	RMSE : 3.19 MAPE : 1.00 errSpread : 0.81 pSpread : 67.81	RMSE : 2.92 MAPE : 1.07 errSpread : 0.84 pSpread : 63.52
t6	RMSE : 4.29 MAPE : 1.44 errSpread : 0.64 pSpread : 60.66	RMSE : 2.96 MAPE : 1.06 errSpread : 0.51 pSpread : 66.65	RMSE : 4.31 MAPE : 1.15 errSpread : 0.52 pSpread : 70.45	RMSE : 4.24 MAPE : 1.13 errSpread : 0.52 pSpread : 70.92	RMSE : 3.95 MAPE : 1.19 errSpread : 0.55 pSpread : 67.17
Avg.	RMSE : 2.52 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 1.92 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.00 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 2.91 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.59 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89

Table 11: Average test metrics for different strike/moneyness levels. The same ranking system is used for coloring as in Tables 9 and 10.

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
itm3	RMSE : 3.06 MAPE : 0.58 errSpread : 0.76 pSpread : 65.49	RMSE : 2.52 MAPE : 0.47 errSpread : 0.62 pSpread : 68.61	RMSE : 4.13 MAPE : 0.68 errSpread : 0.81 pSpread : 69.30	RMSE : 4.11 MAPE : 0.66 errSpread : 0.78 pSpread : 70.82	RMSE : 3.46 MAPE : 0.60 errSpread : 0.74 pSpread : 68.56
itm2	RMSE : 2.74 MAPE : 0.61 errSpread : 0.77 pSpread : 61.32	RMSE : 2.23 MAPE : 0.51 errSpread : 0.67 pSpread : 64.96	RMSE : 3.68 MAPE : 0.71 errSpread : 0.81 pSpread : 65.93	RMSE : 3.58 MAPE : 0.68 errSpread : 0.78 pSpread : 67.00	RMSE : 3.06 MAPE : 0.63 errSpread : 0.76 pSpread : 64.80
itm1	RMSE : 2.79 MAPE : 0.76 errSpread : 0.87 pSpread : 56.57	RMSE : 2.07 MAPE : 0.59 errSpread : 0.71 pSpread : 61.09	RMSE : 3.30 MAPE : 0.78 errSpread : 0.84 pSpread : 63.17	RMSE : 3.19 MAPE : 0.75 errSpread : 0.80 pSpread : 65.12	RMSE : 2.84 MAPE : 0.72 errSpread : 0.81 pSpread : 61.49
atm	RMSE : 2.76 MAPE : 1.00 errSpread : 0.97 pSpread : 51.35	RMSE : 1.91 MAPE : 0.74 errSpread : 0.74 pSpread : 58.29	RMSE : 2.93 MAPE : 0.90 errSpread : 0.84 pSpread : 60.14	RMSE : 2.89 MAPE : 0.88 errSpread : 0.84 pSpread : 60.58	RMSE : 2.63 MAPE : 0.88 errSpread : 0.85 pSpread : 57.59
otm1	RMSE : 2.62 MAPE : 1.46 errSpread : 1.06 pSpread : 44.16	RMSE : 1.85 MAPE : 1.17 errSpread : 0.85 pSpread : 49.48	RMSE : 2.61 MAPE : 1.13 errSpread : 0.83 pSpread : 57.14	RMSE : 2.57 MAPE : 1.10 errSpread : 0.83 pSpread : 58.14	RMSE : 2.41 MAPE : 1.22 errSpread : 0.89 pSpread : 52.23
otm2	RMSE : 2.29 MAPE : 2.51 errSpread : 1.15 pSpread : 40.88	RMSE : 1.62 MAPE : 2.05 errSpread : 0.92 pSpread : 47.62	RMSE : 2.36 MAPE : 1.91 errSpread : 0.90 pSpread : 52.10	RMSE : 2.23 MAPE : 1.86 errSpread : 0.90 pSpread : 51.29	RMSE : 2.12 MAPE : 2.08 errSpread : 0.97 pSpread : 47.97
otm3	RMSE : 2.25 MAPE : 5.27 errSpread : 1.50 pSpread : 31.89	RMSE : 1.68 MAPE : 4.61 errSpread : 1.29 pSpread : 35.69	RMSE : 2.23 MAPE : 3.85 errSpread : 1.11 pSpread : 41.81	RMSE : 2.05 MAPE : 3.47 errSpread : 1.03 pSpread : 44.95	RMSE : 2.05 MAPE : 4.30 errSpread : 1.23 pSpread : 38.59
Avg.	RMSE : 2.64 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 1.98 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.03 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 2.95 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.65 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89

From Tables 10 and 11, it can be seen that, in terms of relative error, the Heston and Bates models produce better predictions for short maturity and OTM options than the B-S and C-S models. Hence, on average, the inclusion of stochastic volatility (SV) and jumps seems to be beneficial for the pricing of these options. This can also be seen in Table 12 which presents the errSpread test values for all pairs of maturity and moneyness levels. However, the opposite is generally true for the RMSE which is more sensitive to outliers. This hints that the Heston and Bates models may occasionally produce the worst price predictions, even though they are accurate on average. This notion is also supported by Table 9, where these models perform the worst during 2020. In the case of long maturity and ITM options, the SV models lose consistently to the C-S model, and occasionally to the B-S model. The RMSE is biased towards these options, so this can also explain why the SV models

have the worst out-of-sample RMSE. Overall, the addition of SV and jumps does not seem as beneficial for the pricing of long maturity and (especially) ITM options.

Table 12: Test errSpread values for different levels of maturity and moneyness. Each cell contains the value for each model separately, and the average value over all models. Here, the model names are abbreviated to the first letter of the last name of each author. The cells are colored based on the average value: smaller values are colored green, and larger values are colored red.

	t1	t2	t3	t4	t5	t6	Avg.
itm3	B-S : 1.09 C-S : 0.95 H : 0.88 B : 0.91 Avg. : 0.96	B-S : 0.62 C-S : 0.48 H : 0.81 B : 0.75 Avg. : 0.66	B-S : 0.70 C-S : 0.50 H : 0.74 B : 0.68 Avg. : 0.66	B-S : 0.80 C-S : 0.61 H : 0.80 B : 0.69 Avg. : 0.72	B-S : 0.78 C-S : 0.67 H : 0.96 B : 0.94 Avg. : 0.84	B-S : 0.59 C-S : 0.53 H : 0.69 B : 0.70 Avg. : 0.63	B-S : 0.76 C-S : 0.62 H : 0.81 B : 0.78 Avg. : 0.74
itm2	B-S : 1.14 C-S : 1.02 H : 0.98 B : 0.96 Avg. : 1.02	B-S : 0.71 C-S : 0.57 H : 0.78 B : 0.81 Avg. : 0.72	B-S : 0.75 C-S : 0.60 H : 0.77 B : 0.73 Avg. : 0.71	B-S : 0.79 C-S : 0.66 H : 0.87 B : 0.76 Avg. : 0.77	B-S : 0.69 C-S : 0.64 H : 0.89 B : 0.81 Avg. : 0.76	B-S : 0.58 C-S : 0.50 H : 0.60 B : 0.59 Avg. : 0.57	B-S : 0.77 C-S : 0.67 H : 0.81 B : 0.78 Avg. : 0.76
itm1	B-S : 1.16 C-S : 0.99 H : 1.11 B : 1.00 Avg. : 1.07	B-S : 0.84 C-S : 0.71 H : 0.88 B : 0.92 Avg. : 0.84	B-S : 0.82 C-S : 0.68 H : 0.81 B : 0.75 Avg. : 0.77	B-S : 0.92 C-S : 0.75 H : 0.89 B : 0.83 Avg. : 0.85	B-S : 0.91 C-S : 0.64 H : 0.86 B : 0.81 Avg. : 0.81	B-S : 0.59 C-S : 0.46 H : 0.50 B : 0.49 Avg. : 0.51	B-S : 0.87 C-S : 0.71 H : 0.84 B : 0.80 Avg. : 0.81
atm	B-S : 1.25 C-S : 0.94 H : 1.21 B : 1.05 Avg. : 1.11	B-S : 0.99 C-S : 0.81 H : 0.95 B : 1.03 Avg. : 0.94	B-S : 0.89 C-S : 0.70 H : 0.78 B : 0.79 Avg. : 0.79	B-S : 0.98 C-S : 0.79 H : 0.89 B : 0.90 Avg. : 0.89	B-S : 1.06 C-S : 0.73 H : 0.77 B : 0.81 Avg. : 0.84	B-S : 0.64 C-S : 0.45 H : 0.45 B : 0.45 Avg. : 0.50	B-S : 0.97 C-S : 0.74 H : 0.84 B : 0.84 Avg. : 0.85
otm1	B-S : 1.58 C-S : 1.29 H : 1.24 B : 1.09 Avg. : 1.30	B-S : 1.12 C-S : 0.94 H : 1.01 B : 1.06 Avg. : 1.03	B-S : 0.90 C-S : 0.77 H : 0.79 B : 0.79 Avg. : 0.81	B-S : 1.05 C-S : 0.82 H : 0.82 B : 0.83 Avg. : 0.88	B-S : 1.05 C-S : 0.78 H : 0.69 B : 0.76 Avg. : 0.82	B-S : 0.65 C-S : 0.51 H : 0.43 B : 0.45 Avg. : 0.51	B-S : 1.06 C-S : 0.85 H : 0.83 B : 0.83 Avg. : 0.89
otm2	B-S : 2.18 C-S : 1.81 H : 1.56 B : 1.49 Avg. : 1.76	B-S : 1.22 C-S : 0.99 H : 1.07 B : 1.03 Avg. : 1.08	B-S : 0.91 C-S : 0.74 H : 0.84 B : 0.89 Avg. : 0.85	B-S : 0.97 C-S : 0.72 H : 0.79 B : 0.83 Avg. : 0.83	B-S : 0.97 C-S : 0.75 H : 0.68 B : 0.71 Avg. : 0.78	B-S : 0.64 C-S : 0.51 H : 0.45 B : 0.42 Avg. : 0.51	B-S : 1.15 C-S : 0.92 H : 0.90 B : 0.90 Avg. : 0.97
otm3	B-S : 2.26 C-S : 2.01 H : 1.65 B : 1.56 Avg. : 1.87	B-S : 1.73 C-S : 1.42 H : 1.43 B : 1.38 Avg. : 1.49	B-S : 1.60 C-S : 1.34 H : 1.14 B : 1.02 Avg. : 1.28	B-S : 1.43 C-S : 1.21 H : 1.00 B : 0.84 Avg. : 1.12	B-S : 1.24 C-S : 1.12 H : 0.92 B : 0.84 Avg. : 1.03	B-S : 0.77 C-S : 0.63 H : 0.53 B : 0.52 Avg. : 0.61	B-S : 1.50 C-S : 1.29 H : 1.11 B : 1.03 Avg. : 1.23
Avg.	B-S : 1.52 C-S : 1.29 H : 1.23 B : 1.15 Avg. : 1.30	B-S : 1.03 C-S : 0.84 H : 0.99 B : 1.00 Avg. : 0.97	B-S : 0.94 C-S : 0.76 H : 0.84 B : 0.81 Avg. : 0.84	B-S : 0.99 C-S : 0.79 H : 0.87 B : 0.81 Avg. : 0.87	B-S : 0.96 C-S : 0.76 H : 0.83 B : 0.81 Avg. : 0.84	B-S : 0.64 C-S : 0.51 H : 0.52 B : 0.52 Avg. : 0.55	B-S : 1.01 C-S : 0.83 H : 0.88 B : 0.85 Avg. : 0.89

From Table 12, it is clear that the errSpread metric is not constant across different maturities and strikes. In particular, the average values are larger for short maturity and OTM options, which is at least partly caused by the loss used during training. Section 4.4 discusses more about this.

Finally, we investigate the effect of the random training process on the performance of the models. The best hyperparameter version of each model is run with 12 different random seeds, which provides some (albeit small) amount of statistics. Figure 16 shows the effect of the random seed on the average metrics for different models and spilt types (train,validation,test). With the exception of the MAPE, the validation and test metrics are worse than the train metrics, as also shown in Table 8. Moreover, the validation and test metrics tend to vary more than the train metrics, especially in the case of the RMSE and errSpread. The C-S model seems to be the least sensitive, and the Heston model the most sensitive, to the random training process. Overall, the effect of the random seed is definitely not insignificant. For instance, the test errSpread metric for the Heston model varies between 0.60 and 1.30 which is a very wide range. In fact, the Heston model produces the best validation and test results for a single seed in three categories, as shown by the boxplot whiskers in Figure 16.

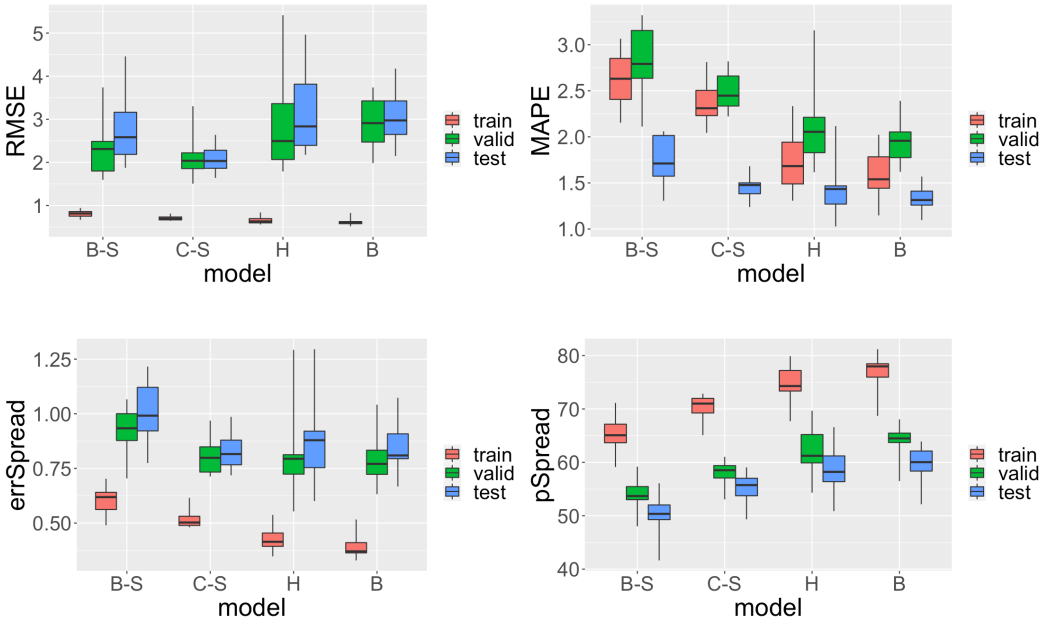

Figure 16: Boxplots for different metrics and models, separately for training, validation and test intervals. For each model and split type, the box describes (from bottom to top) the minimum, the first quartile, the median, the third quartile and the maximum of the values over all random seeds. For each seed, the values are averaged over all time splits.

Now, the question is how to choose the best version of each model from the different seeds. Table 13 shows the correlations of the metrics over the seeds between different split types. From the table, it can be seen that the correlations between the training and the validation intervals, and the training and the test intervals, are clearly positive on average. This suggests that a suitable strategy would be to simply choose the model version (seed) with the best train metrics. On the other hand, the correlations between the validation and the test sets are more strongly positive. Therefore, before deploying the models in real time, it could be worthwhile to evaluate the versions of each model on a short validation interval, and choose the version with the best validation metrics.

Table 13: Correlations of the metrics over the random seeds between (a) the train and the validation intervals, (b) the train and the test intervals, and (c) the validation and the test intervals.

(a) train-validation

Model	RMSE	MAPE	errSpread	pSpread	Avg.
B-S	0.203	0.950	0.766	0.963	0.721
C-S	0.343	0.691	0.604	0.886	0.631
H	0.298	0.808	0.677	0.960	0.686
B	-0.214	0.840	0.618	0.951	0.549
Avg.	0.157	0.822	0.666	0.940	0.646

(b) train-test

Model	RMSE	MAPE	errSpread	pSpread	Avg.
B-S	0.188	0.732	0.629	0.967	0.629
C-S	0.340	0.545	0.579	0.883	0.587
H	0.132	0.495	0.551	0.944	0.531
B	-0.237	0.207	0.370	0.924	0.316
Avg.	0.106	0.495	0.533	0.929	0.516

(c) validation-test

Model	RMSE	MAPE	errSpread	pSpread	Avg.
B-S	0.472	0.843	0.780	0.973	0.767
C-S	0.808	0.712	0.841	0.875	0.809
H	0.885	0.839	0.942	0.964	0.907
B	0.921	0.447	0.848	0.965	0.795
Avg.	0.772	0.710	0.853	0.944	0.820

4.3 Model Parameters

In this section, we present the parameters of the models. As before, we only consider call options and the best hyperparameters of each model. Table 14 lists the parameter averages over all contracts and time splits, and the standard deviations over the (12) random seeds, separately for train, validation and test intervals. The volatility parameters σ, $\sqrt{\theta}$ and $\sqrt{V_0}$ have similar values in the case of all models, which not surprising. In the case of the C-S model, the skewness and the kurtosis deviate slightly from those of the normal distribution. Still, even the small amounts of negative skewness and positive excess kurtosis seem to have a positive impact on the prediction performance of the C-S model, as shown in Section 4.2. For the Heston and Bates models, the five mutual parameters (κ,θ,V_0,ω,ρ) are quite similar, but there are also small differences. Most notably, the volatility of volatility ω is on average larger in the case of the Bates model. Additionally, the initial volatility $\sqrt{V_0}$ is slightly larger in the case of the Heston model. Finally, the averages of all the Bates jump parameters lie somewhere in the middle of the ranges defined in Table 2.

By Table 14, most of the parameters have a relatively large standard deviations relative to the average values. The most notable exception is the B-S volatility which varies very little between different seeds. This is not surprising, because the B-S model has only a single parameter, so the volatility cannot be varied much without weakening the pricing accuracy. Additionally, the C-S volatility varies noticeably less than the volatility parameters of the SV models. The large variations of the remaining parameters can be explained by the fact that the corresponding models have multiple parameters. Thus, different combinations of the parameters can produce similar price estimates. Consequently, it is not advisable to examine the parameters of a model separately, as only the combined effect of the parameters can be considered meaningful.

Table 14: Average parameter values for train,validation and test intervals. For each parameter, the percentage standard deviation (relative to the average value) over the random seeds is listed after the '±' symbol.

Param.	Black-Scholes	Corrado-Su	Heston	Bates
σ	train : 0.162 ± 0.0% valid : 0.178 ± 0.5% test : 0.191 ± 0.5%	train : 0.166 ± 2.3% valid : 0.183 ± 2.7% test : 0.195 ± 2.9%		
μ_3		train : -0.151 ± 40.8% valid : -0.153 ± 45.4% test : -0.152 ± 46.3%		
μ_4		train : 3.529 ± 15.5% valid : 3.522 ± 15.5% test : 3.535 ± 17.0%		
κ			train : 4.312 ± 11.7% valid : 4.358 ± 11.3% test : 4.426 ± 10.1%	train : 4.281 ± 10.1% valid : 4.331 ± 9.6% test : 4.404 ± 8.4%
$\sqrt{\theta}$			train : 0.179 ± 7.8% valid : 0.188 ± 8.3% test : 0.195 ± 8.4%	train : 0.173 ± 9.1% valid : 0.182 ± 9.6% test : 0.189 ± 9.7%
$\sqrt{V_0}$			train : 0.166 ± 9.0% valid : 0.187 ± 8.9% test : 0.202 ± 8.3%	train : 0.156 ± 9.9% valid : 0.176 ± 9.9% test : 0.191 ± 9.4%
ω			train : 0.452 ± 29.6% valid : 0.457 ± 28.3% test : 0.463 ± 26.6%	train : 0.506 ± 29.4% valid : 0.509 ± 27.8% test : 0.514 ± 26.5%
ρ			train : -0.339 ± 51.8% valid : -0.330 ± 55.0% test : -0.318 ± 59.5%	train : -0.339 ± 41.8% valid : -0.331 ± 44.8% test : -0.317 ± 49.7%
λ_J				train : 6.965 ± 32.6% valid : 7.218 ± 32.5% test : 7.399 ± 32.1%
μ_J				train : -0.013 ± 38.3% valid : -0.013 ± 38.6% test : -0.013 ± 37.6%
σ_J				train : 0.010 ± 42.2% valid : 0.011 ± 40.5% test : 0.011 ± 40.7%

Tables 15, 16 and 17 show the average parameter values for different years, maturities and strikes, respectively. From the tables, it is evident that a single set of parameters for all time steps, maturities and strikes is not sufficient in the case of any model. By Table 15 for instance, higher values of the volatility parameters (σ, θ and $\sqrt{V_0}$) are required during 2020 and 2022 than during the other test years. Moreover, in the case of call options, the average volatilities increase with maturity and moneyness, as seen in Tables 16 and 17. This well-known phenomenon is known as the *volatility smile*. Figure 17 shows an example of the B-S volatility smile, visualized in three dimensions.

Table 15: Average parameter values for each test year.

Param.	2018	2019	2020	2021	2022
σ	B-S : 0.151 C-S : 0.153	B-S : 0.148 C-S : 0.150	B-S : 0.240 C-S : 0.244	B-S : 0.178 C-S : 0.185	B-S : 0.234 C-S : 0.243
μ_3	-0.095	-0.114	-0.094	-0.205	-0.244
μ_4	3.409	3.249	3.465	3.732	3.806
κ	H : 4.239 B : 4.209	H : 4.212 B : 4.145	H : 4.500 B : 4.424	H : 4.591 B : 4.627	H : 4.600 B : 4.626
$\sqrt{\theta}$	H : 0.171 B : 0.163	H : 0.173 B : 0.171	H : 0.228 B : 0.218	H : 0.188 B : 0.181	H : 0.216 B : 0.211
$\sqrt{V_0}$	H : 0.155 B : 0.144	H : 0.149 B : 0.135	H : 0.257 B : 0.250	H : 0.192 B : 0.179	H : 0.255 B : 0.245
ω	H : 0.382 B : 0.477	H : 0.383 B : 0.450	H : 0.427 B : 0.482	H : 0.565 B : 0.577	H : 0.559 B : 0.583
ρ	H : -0.278 B : -0.306	H : -0.285 B : -0.353	H : -0.367 B : -0.328	H : -0.333 B : -0.312	H : -0.335 B : -0.291
λ_J	7.329	7.434	8.283	6.761	7.207
μ_J	-0.014	-0.014	-0.016	-0.011	-0.012
σ_J	0.011	0.011	0.013	0.011	0.01

Table 16: Average parameter values for different maturity levels.

Param.	t1	t2	t3	t4	t5	t6
σ	B-S : 0.177 C-S : 0.180	B-S : 0.186 C-S : 0.189	B-S : 0.190 C-S : 0.194	B-S : 0.193 C-S : 0.198	B-S : 0.197 C-S : 0.204	B-S : 0.200 C-S : 0.208
μ_3	-0.122	-0.129	-0.143	-0.155	-0.173	-0.188
μ_4	3.358	3.413	3.48	3.543	3.657	3.761
κ	H : 4.287 B : 4.226	H : 4.415 B : 4.368	H : 4.451 B : 4.417	H : 4.469 B : 4.449	H : 4.475 B : 4.483	H : 4.458 B : 4.482
$\sqrt{\theta}$	H : 0.183 B : 0.176	H : 0.194 B : 0.187	H : 0.198 B : 0.191	H : 0.199 B : 0.193	H : 0.199 B : 0.193	H : 0.200 B : 0.193
$\sqrt{V_0}$	H : 0.190 B : 0.182	H : 0.195 B : 0.186	H : 0.198 B : 0.189	H : 0.202 B : 0.191	H : 0.210 B : 0.196	H : 0.219 B : 0.204
ω	H : 0.454 B : 0.503	H : 0.449 B : 0.503	H : 0.454 B : 0.508	H : 0.460 B : 0.514	H : 0.474 B : 0.524	H : 0.486 B : 0.533
ρ	H : -0.411 B : -0.431	H : -0.349 B : -0.360	H : -0.321 B : -0.325	H : -0.300 B : -0.297	H : -0.273 B : -0.256	H : -0.255 B : -0.232
λ_J	6.696	7.259	7.457	7.574	7.684	7.725
μ_J	-0.012	-0.013	-0.014	-0.014	-0.014	-0.014
σ_J	0.01	0.011	0.012	0.012	0.012	0.012

Table 17: Average parameter values for different moneyness levels.

Param.	itm3	itm2	itm1	atm	otm1	otm2	otm3
σ	B-S : 0.236 C-S : 0.241	B-S : 0.220 C-S : 0.226	B-S : 0.205 C-S : 0.210	B-S : 0.189 C-S : 0.194	B-S : 0.173 C-S : 0.178	B-S : 0.160 C-S : 0.164	B-S : 0.151 C-S : 0.154
μ_3	-0.21	-0.186	-0.163	-0.145	-0.131	-0.118	-0.11
μ_4	3.849	3.711	3.596	3.506	3.429	3.356	3.301
κ	H : 4.495 B : 4.473	H : 4.514 B : 4.503	H : 4.511 B : 4.509	H : 4.482 B : 4.477	H : 4.416 B : 4.396	H : 4.318 B : 4.279	H : 4.244 B : 4.193
$\sqrt{\theta}$	H : 0.221 B : 0.216	H : 0.213 B : 0.206	H : 0.202 B : 0.195	H : 0.192 B : 0.184	H : 0.183 B : 0.175	H : 0.178 B : 0.172	H : 0.180 B : 0.175
$\sqrt{V_0}$	H : 0.262 B : 0.249	H : 0.237 B : 0.224	H : 0.214 B : 0.201	H : 0.194 B : 0.182	H : 0.177 B : 0.166	H : 0.167 B : 0.158	H : 0.166 B : 0.159
ω	H : 0.423 B : 0.446	H : 0.427 B : 0.461	H : 0.431 B : 0.479	H : 0.439 B : 0.500	H : 0.460 B : 0.528	H : 0.503 B : 0.569	H : 0.555 B : 0.618
ρ	H : -0.183 B : -0.147	H : -0.210 B : -0.178	H : -0.244 B : -0.218	H : -0.286 B : -0.272	H : -0.345 B : -0.355	H : -0.431 B : -0.469	H : -0.530 B : -0.578
λ_J	7.935	7.838	7.697	7.508	7.271	6.977	6.569
μ_J	-0.014	-0.014	-0.014	-0.014	-0.013	-0.013	-0.013
σ_J	0.012	0.012	0.012	0.011	0.011	0.01	0.011

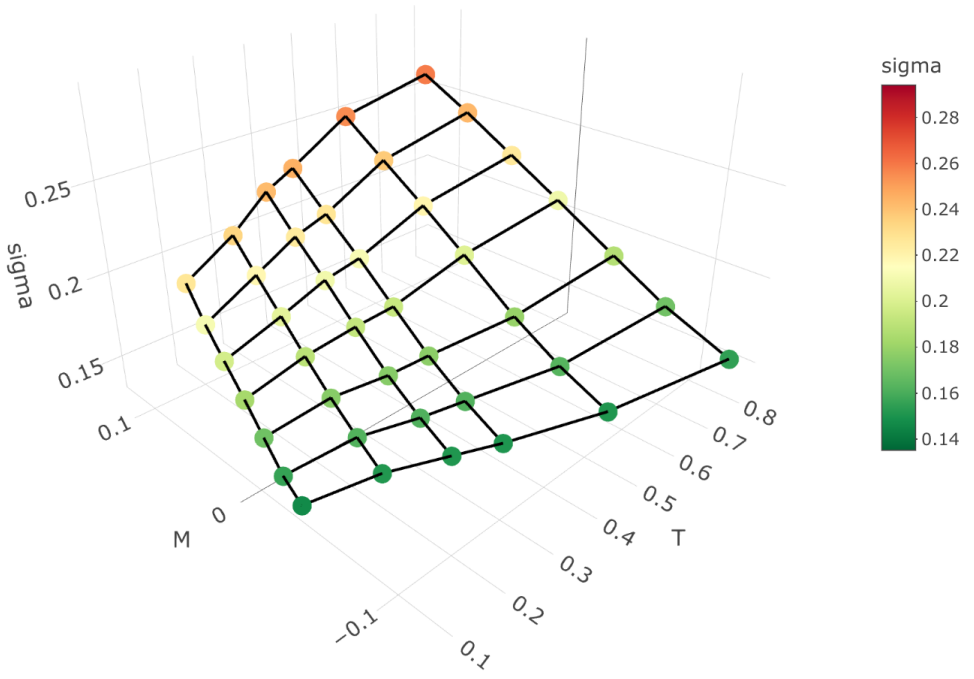

Figure 17: Black-Scholes volatility surface in the afternoon of 2022-02-01 as a function of the time to maturity T (in years) and the moneyness M.

In Figure 18, the average short maturity volatilities are compared to Cboe's Volatility Index (VIX) which is calculated from the prices of 30-day constant maturity SPX options [Cboe, 2022]. The VIX index is a useful baseline, because it is often used as a proxy for the expected market volatility. Here,

we use only a single random seed for each model, and the values are computed for the aggregate test period. As can be seen from the figure, the average volatility parameters and the VIX move very similarly. This is a desirable result, for different behaviours would raise suspicions about the reliability of the models. However, the VIX tends to exhibit slightly larger values for the majority of the test period. This may be due to the fact that VIX is not calculated in the same way as the implied volatilities of the parametric models (in fact, VIX is computed without any model asssumptions; see [Cboe, 2022] for more details). Additionally, VIX is calculated from options that are not identical to the ones used in this thesis.

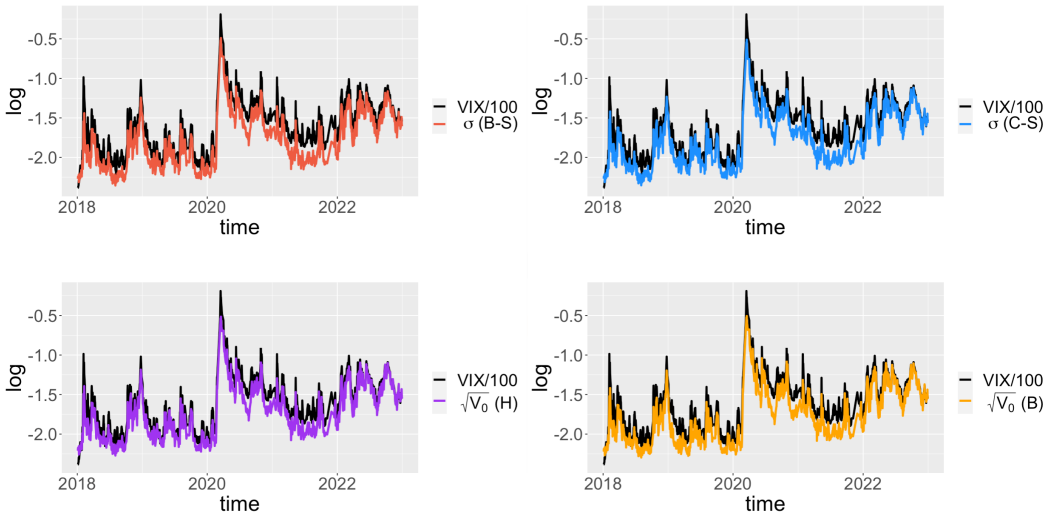

Figure 18: The volatility parameters σ (Black-Scholes,Corrado-Su) and $\sqrt{V_0}$ (Heston,Bates) from 2018 to 2022, averaged over different contract strikes. Here, a logarithmic scale is used. The average parameter values are compared to the VIX index, shown in black. Since the VIX index is calculated using options with a maturity of 30 days, the volatility parameters are only computed using the contracts of the first maturity level (`t1`).

Finally, we demonstrate the effect of the model parameters on the implied log return distribution of the underlying asset. For each model, we draw samples using the techniques presented in Chapter 2. Additionally, the samples are drawn from the risk-neutral densities, so the risk aversion coefficient γ from Equation (31) is set to zero. Furthermore, we assume that the expected rate of return is zero, which can be a reasonable approximation for short time horizons. Figure 19 shows 10-day log returns that have been sampled from the distributions determined by the average short maturity (`t1`) model parameters. By Figure 19(a), the C-S, Heston and Bates implied distributions are visibly different from the normal distribution given by the B-S model.

In particular, the C-S, Heston and Bates distributions all have nonnegative skewness (-0.26, -0.43 and -0.61, respectively) and positive excess kurtosis (0.63, 0.28 and 0.66, respectively). The effect of this can be seen in Figures 19(b) and 19(c), where the tails of the three aforementioned models are thicker on the left, but thinner on the right, compared to the B-S model. Hence, the three distributions are more realistic than the B-S distribution, in the sense that the empirical short-term log returns are known to have negative skewness and positive excess kurtosis (see Figure 3). For example, if the implied distributions are used as an aid in risk management, the fatter left tails can provide more realistic and conservative estimates of the possible losses.

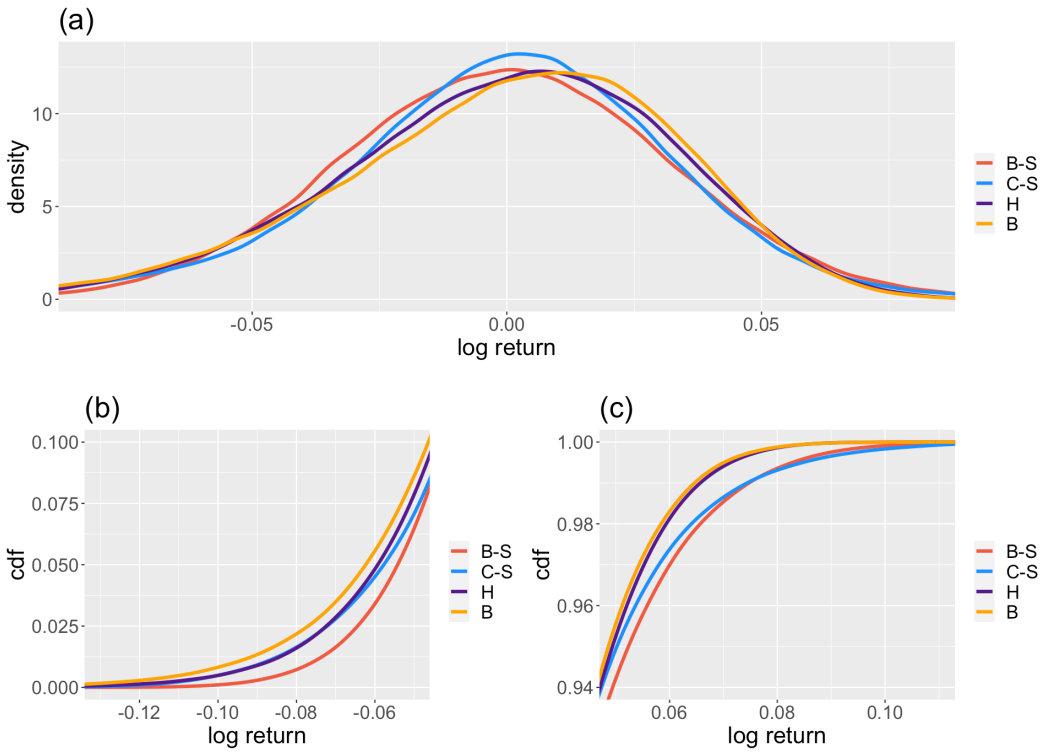

Figure 19: Implied 10-day log return distributions sampled from the average model parameters of the `t1` maturity level options in the afternoon of 2022-04-01. For each model, $N = 2 \cdot 10^5$ log returns are drawn from the implied (risk-neutral) distribution. Here, we assume an expected rate of return of zero. The subplots show (a) the empirical log return density, and (b) the left and (c) the right tails of the empirical cumulative distribution function.

4.4 Discussion

Overall, the results from Section 4.2 are promising, but the models still have some limitations in their current form. First, the models exhibit some degree of overfitting, as shown by Table 8 and Figure 16. This could be mitigated using standard regularization techniques for neural networks, such as weight regularization (either L1 or L2), or dropout. These techniques are not investigated in this thesis, but they could be beneficial, especially for larger networks. However, the use of regularization has some caveats. First, one must choose suitable regularization hyperparameters, such as the weight regularization (L1 and L2) coefficients and the dropout rate, which increases the computational burden. Second, the addition of regularization can make it challenging to obtain a good in-sample fit during training. By Table 13, train performance correlates positively with the validation and test performances, so a good fit to the training data is important. Finally, the regularization techniques do not guarantee a better out-of-sample performance.

In addition to slight overfitting, the choice of the loss function could also be scrutinized. Here, we use the loss in Equation (63) which partly compensates for the MSE bias (in terms of relative error) towards more expensive options. However, this choice does not eliminate the bias, as seen in Table 12. We select the loss with the inverse square root prices $y_i^{-1/2}$ as weights, because this loss seems to be more well-behaved during training than the losses with either y_i^{-1} or y_i^{-2} as weights. Good results may be achieved with the latter weights, but this may require more supervision during training (e.g., custom learning rates and number of epochs for different models and/or training intervals). Nevertheless, a different loss function or weighting scheme may be required when fitting the models to very short maturity (less than a month) options, in which case the MSE bias would be even worse. Alternatively, to reduce the bias, the models could be fitted to only short term and/or OTM options. However, multiple versions of each model would then be needed to fit the entire maturity-moneyness grid.

One limitation of the models is that the parameter ranges have to be fixed beforehand when using the sigmoid activation functions. On one hand, this results in faster training, and prevents unreasonable parameter values, even out-of-sample. On the other hand, the models cannot produce parameter values outside the predetermined ranges, even when the market prices would imply such values. For example, if the implied volatility of the options were to rise above 0.8, or shrink below 0.05, the models would not be able to produce accurate price estimates when using the ranges in Table 2. Moreover, the models will likely have trouble extrapolating the parameter values outside of the training set ranges. For instance, if the maximum implied volatility in the training set was σ_0, the models would likely struggle in a test set containing several implied volatilities above σ_0. Indeed, this type of problem can be

observed during the test year 2020 (see the third row in Table 9). Therefore, in practice, it is advisable to include wide ranges of parameters (i.e., different types of market conditions) in the training set, so that possible extrapolation is kept at a minimum. Due to the possible performance decay of the models with time (see Table 8 and Figure 16), it could also be beneficial to either (i) omit the validation period entirely, or (ii) include a validation period shorter than the 5-month long period used in Section 4.2. For the same reason, the 14-day gap between the train and validation/test intervals (that was used in Section 4.2) should be removed. As Figure 16 and Table 13 suggest, it is recommended to train the models with multiple random seeds, and then select the version with either the best train performance (if validation period is omitted) or the best validation performance.

With the exception of the Black-Scholes model, the option pricing models studied have multiple parameters, which presents some challenges. First of all, this can complicate the training process, as it is more difficult to find a good local minimum for multiple parameters simultaneously. Moreover, the resulting parameters are highly sensitive to the randomness of the training process, and the initial network weights in particular, as seen in Table 14. This especially affects the Bates and Heston models, because these models have the largest number of parameters, and they seem to be the hardest to fit to the training data. As in the previous paragraph, the randomness can be taken into account by repeating the training process for multiple random seeds. Additionally, it is possible to fix a subset of the parameters before training, which can reduce the variation in the remaining free parameters.

In Section 4.1, we select estimates for the risk-free rate and the dividend yield of the S&P500 index, and these estimates obviously have an effect on the final results. For the risk-free rate, we use US Treasury Security yields of constant maturities, and interpolate these yields using the cubic spline interpolation. Alternatively, we could have used different proxies for the risk-free rate, such as rates for interest rate swaps. Moreover, the choice of the interpolation scheme affects the risk-free rate estimates in between the constant maturities. For dividends, we use a pre-calculated yield from GuruFocus. In reality however, the dividends of the companies in the S&P500 index are discrete, which means that the continuous yield approximation can be rather crude, especially for short maturity options. Instead of using a continuous yield, one could adjust the underlying asset price for discrete dividends, which could result in more accurate price estimates. However, the use of the continuous yield is more convenient, and in the case of the S&P500 index, the discrete dividends of a single company do not have as much of an impact.

The work of this thesis could be continued in several ways. For example, the existing models could be applied to other underlying assets, such as other indices, or individual stocks. Additionally, the inverse map framework could be

applied to other option pricing models. The neural network approach may limit the choice of models (see the end of Section 3.4), but the models considered here are certainly not the only applicable ones. Furthermore, one could also investigate the use of larger networks, or even different network types. Table 7 suggests that the validation error can be lowered even more by increasing the number of hidden layer units and/or layers. For the larger networks in particular, it could also be beneficial to study the effects of regularization. On the other hand, regularization techniques could also be applied to the existing networks in order to check whether the observed overfitting could be mitigated.

In Section 3.3, multiple choices are made regarding the network structure and the training process. Therefore, these choices could be modified to further study their effect on the results. For example, the network inputs defined in (62) could be changed. This would be especially interesting in the case where calls and puts are fitted simultaneously. It is also possible that larger networks are required when both calls and puts are used. It should be emphasized that many of the choices made in Section 3.3, including the final sigmoid ranges for the model parameters, can be suboptimal for underlying assets other than the SPX. Therefore, some caution should be exercised when fitting the models to the options of other assets.

An interesting consideration is the application of the inverse map framework to American options. Although exact pricing formulas for these options are not available, there exist approximations, such as the one presented by Barone-Adesi and Whaley [1987]. Moreover, the value of an American options can be decomposed into two parts: the value of the corresponding European option, and the early exercise premium. The former part can be determined using European option pricing formulas, while the latter part can be estimated in a nonparametric fashion.

Finally, the usefulness of the model outputs could be further investigated. For instance, each set of model parameters, such as (σ, μ_3, μ_4) in the case of the Corrado-Su model, implies a certain risk-neutral distribution for the underlying asset. It is not clear however, how informative these distributions are in practice. Before assessing that, the risk-neutral distribution should be converted to the 'real' distribution, which corresponds to a change of probability measure from $\mathbb{Q}$ to $\mathbb{P}$ (see Equation (12)). For the Heston and Bates models, this means that the risk-neutral parameters should be converted to the real ones using Equations (32) and (51). Additionally, in the case of all models, the risk-free rate r should be replaced by the real expected rate of return μ. Analogously to the capital asset pricing model, this can be done with the transformation $\mu = r + \lambda\sigma$, where σ is the current volatility (σ for Black-Scholes and Corrado-Su, $\sqrt{V_0}$ for Heston and Bates), and λ is the market price of risk (*not* equal to the price of volatility risk λ from Equation (30)).

In addition to the implied distributions, the models can also compute the Greeks of the options. In a neural network setting, this is straightforward thanks to automatic differentiation. The most important application for the Greeks is hedging. In a typical scenario, a call option is sold, and shares of the underlying asset are bought to hedge the option position. The goal is to minimize the variance of the portfolio consisting of the option and the underlying asset. The quality of the hedge can then be assessed by computing the realized portfolio variance over multiple time steps. By doing this for multiple option pricing models, one can then determine which model produces the best hedging strategy.[1] However, the examination of hedging performances is omitted in this thesis.

[1]Note that under the Bates model, a theoretically perfect hedge is not possible due to the jump component in the underlying price process.

5 Conclusions

Many tasks in finance, such as hedging, or calculation of implied asset return distributions, require accurate modeling of option prices observed in the market. For this, many parametric pricing models have been proposed in the literature. Most models have been developed for European options, because exact closed-form solutions can be found for these options. To estimate their parameters, some error is typically minimized between the model prices and the market prices. This estimation process is called model calibration.

In this thesis, the option pricing models considered are (i) the Black-Scholes (B-S) model, (ii) the Corrado-Su (C-S) model, (iii) the Heston model and (iv) the Bates model. The Black-Scholes model is the simplest, as it assumes a constant volatility and log-normal returns for the underlying asset. The remaining models extend the B-S model to allow nonzero skewness and excess kurtosis for the log return distribution. Additionally, the Heston and Bates models treat the asset volatility as a random variable, and the Bates model adds jumps to the underlying price process.

The goal of this thesis is to calibrate the above option pricing models using neural networks. That is, the aim is find the unobservable model parameters that produce the best fit to the market prices. In a neural network setting, the network takes in market variables as inputs, and outputs the model parameters. These parameters are then passed to the corresponding pricing formula, and the resulting prices are compared to the market prices by evaluating some loss function. Using the gradients of the loss with respect to the network weights, the parameter values can be updated until a satisfactory solution is found. In this thesis, the above calibration framework is referred to as the inverse map approach.

The inverse map approach has several advantages. First, rather than directly estimating the option prices using neural networks, the final price predictions are given by the parametric models. This is beneficial, because the parametric models are (i) more interpretable and (ii) arbitrage-free by construction. Due to the second property, separate arbitrage restrictions need not be imposed during training. Second, compared to traditional calibration methods, where the goal is to only find the parameter values, one also learns the inverse pricing formula. Thus, after the calibration process, the model can price future observations without the need to re-calibrate at every time step. Finally, due to the automatic differentiation ability of neural networks, it is straightforward to calculate the Greeks of the option pricing models.

In this thesis, we apply the inverse map approach to the European SPX options from 2015 to 2022. To evaluate the models, multiple train-validation-test splits are used, and the evaluation metrics are averaged over these splits. For evaluation, we use four metrics that assess different aspects of the model price predictions. Moreover, to account for the randomness in the training process, we run each model and time split for multiple random seeds. The

validation splits are used to select the hyperparameters that control the size of the neural network (number of layers and units), and the test intervals are used to evaluate the best hyperparameters of each model.

Section 4.2 presents the model evaluation results that are obtained using only call options. In the case of all models, the largest allowed network (2 hidden layers and 10 units per hidden layer) produces the best in-sample and out-of-sample results. Moreover, the models are able to produce good overall results. For the best version (best hyperparameters and seed) of each model, the average pricing error relative to the option bid-ask spread is well below one for both validation and test intervals. Furthermore, more than half of the out-of-sample predictions land inside the bid-ask spread in the case of all models. However, we find that such results are achieved only when the models are fitted to calls and puts separately. For the train intervals, the rank order of the models is (1) Bates, (2) Heston, (3) C-S, (4) B-S. That is, more complex models achieve better train metrics. For the validation and test intervals however, the order is not as clear. Overall, the Bates and C-S models seem to have the strongest, and the B-S model the weakest, validation and test results, when the metrics are averaged over all random seeds. However, the Heston model has the best validation and test metrics for a single seed. Overall, the results for puts are slightly better than the above results for calls (see Appendix A).

The performance of the models is not identical for different time splits, maturities and strikes. In particular, the models have the worst out-of-sample performance during the year 2020. Additionally, relative pricing errors are larger for short maturity and OTM options than they are for long maturity and ITM options. The varying percentage errors (MAPE) between different strike prices are especially noticeable for call options. By training the models for multiple random seeds, we observe that on average, there is a clear positive correlation between the train, validation and test metrics. Therefore, train and validation metrics seem to be good indicators for future performance of the models.

In addition to the pricing performance, we study the parameters produced by the models. We find that some parameter values can vary significantly between different random seeds. Thus, it may not be meaningful to interpret the parameters of a model separately. Moreover, the average parameter values vary between different time periods, maturities and strikes, as expected. We also find that the average volatility parameters behave very similarly to the VIX index throughout the aggregate test interval.

In future work, the inverse map approach could be applied to other underlying assets and option pricing models, and even other option types. Additionally, the neural network, and the associated training process, could be modified in order to find better results. Finally, it would be interesting to investigate the usefulness of the implied return distributions and hedging strategies produced by the models.

References

Andreou, P. C., Charalambous, C. and Martzoukos, S. H. [2010]. Generalized parameter functions for option pricing, *Journal of Banking & Finance* **34**(3): 633–646.

Barone-Adesi, G. and Whaley, R. E. [1987]. Efficient analytic approximation of American option values, *the Journal of Finance* **42**(2): 301–320.

Bates, D. S. [1996]. Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options, *The Review of Financial Studies* **9**(1): 69–107.

Baxter, M., Rennie, A. and Rennie, A. J. [1996]. *Financial Calculus: An Introduction to Derivative Pricing*, Cambridge University Press.

Black, F. and Scholes, M. [1973]. The pricing of options and corporate liabilities, *Journal of Political Economy* **81**(3): 637–654.

Bollerslev, T., Gibson, M. and Zhou, H. [2011]. Dynamic estimation of volatility risk premia and investor risk aversion from option-implied and realized volatilities, *Journal of Econometrics* **160**(1): 235–245.

Borland, L. [2002]. Option pricing formulas based on a non-gaussian stock price model, *Physical Review Letters* **89**(9): 098701.

Borland, L. and Bouchaud, J.-P. [2004]. A non-gaussian option pricing model with skew, *Quantitative Finance* **4**(5): 499–514.

Brown, C. A. and Robinson, D. M. [2002]. Skewness and kurtosis implied by option prices: A correction, *Journal of Financial Research* **25**(2): 279–282.

Cboe [2022]. Volatility Index Methodology, `https://cdn.cboe.com/api/global/us_indices/governance/Volatility_Index_Methodology_Cboe_Volatility_Index.pdf`. Accessed: 2023-03-13.

Cboe [2023]. Cboe DataShop, `https://datashop.cboe.com`. Accessed: 2023-01-30.

Corrado, C. J. and Su, T. [1996]. Skewness and kurtosis in S&P 500 index returns implied by option prices, *Journal of Financial Research* **19**(2): 175–192.

Cox, J. C. [1997]. The Constant Elasticity of Variance Option Pricing Model, *The Journal of Portfolio Management* **23**(5): 15–17.

Cui, Y., del Baño Rollin, S. and Germano, G. [2017]. Full and fast calibration of the Heston stochastic volatility model, *European Journal of Operational Research* **263**(2): 625–638.

Date, P. and Islyaev, S. [2015]. A fast calibrating volatility model for option pricing, *European Journal of Operational Research* **243**(2): 599–606.

Duffie, D., Pan, J. and Singleton, K. [2000]. Transform analysis and asset pricing for affine jump-diffusions, *Econometrica* **68**(6): 1343–1376.

Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C. and Garcia, R. [2009]. Incorporating functional knowledge in neural networks, *Journal of Machine Learning Research* **10**(6): 1239–1262.

FRED [2023]. Federal Reserve Economic Data, `https://fred.stlouisfed.org`. Accessed: 2023-01-30.

Garcia, R. and Gençay, R. [2000]. Pricing and hedging derivative securities with neural networks and a homogeneity hint, *Journal of Econometrics* **94**(1-2): 93–115.

Glorot, X. and Bengio, Y. [2010]. Understanding the difficulty of training deep feedforward neural networks, *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, pp. 249–256.

GuruFocus [2023]. S&P 500 Dividend Yield, `https://www.gurufocus.com/economic_indicators/150/sp-500-dividend-yield`. Accessed: 2023-02-09.

Heston, S. L. [1993]. A closed-form solution for options with stochastic volatility with applications to bond and currency options, *The Review of Financial Studies* **6**(2): 327–343.

Hornik, K., Stinchcombe, M. and White, H. [1989]. Multilayer feedforward networks are universal approximators, *Neural Networks* **2**(5): 359–366.

Hull, J. and White, A. [1987]. The pricing of options on assets with stochastic volatilities, *The Journal of Finance* **42**(2): 281–300.

Hutchinson, J. M., Lo, A. W. and Poggio, T. [1994]. A nonparametric approach to pricing and hedging derivative securities via learning networks, *The Journal of Finance* **49**(3): 851–889.

Itkin, A. [2019]. Deep learning calibration of option pricing models: Some pitfalls and solutions, *arXiv preprint arXiv:1906.03507* .

Itô, K. [1951]. *On Stochastic Differential Equations*, number 4, Memoirs, American Mathematical Society.

Jondeau, E. and Rockinger, M. [2001]. Gram–Charlier densities, *Journal of Economic Dynamics and Control* **25**(10): 1457–1483.

Jurczenko, E., Maillet, B. and Negréa, B. [2004]. A note on skewness and kurtosis adjusted option pricing models under the martingale restriction, *Quantitative Finance* **4**(5): 479–488.

Kahl, C. and Jäckel, P. [2005]. Not-so-complex logarithms in the Heston model, *Wilmott Magazine* **19**(9): 94–103.

Kingma, D. P. and Ba, J. [2014]. Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* . Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

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

León, Á., Mencía, J. and Sentana, E. [2009]. Parametric properties of semi-nonparametric distributions, with applications to option valuation, *Journal of Business & Economic Statistics* **27**(2): 176–192.

Lord, R., Koekkoek, R. and Dijk, D. V. [2010]. A comparison of biased simulation schemes for stochastic volatility models, *Quantitative Finance* **10**(2): 177–194.

Madan, D. B., Carr, P. P. and Chang, E. C. [1998]. The variance gamma process and option pricing, *Review of Finance* **2**(1): 79–105.

Merton, R. C. [1976]. Option pricing when underlying stock returns are discontinuous, *Journal of Financial Economics* **3**(1-2): 125–144.

Mitra, S. [2011]. A review of volatility and option pricing, *International Journal of Financial Markets and Derivatives* **2**(3): 149–179.

OCC [2023]. Historical volume statistics, `https://www.theocc.com/market-data/market-data-reports/volume-and-open-interest/historical-volume-statistics`. Accessed: 2023-01-28.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P. and Ommer, B. [2022]. High-resolution image synthesis with latent diffusion models, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695.

Ruf, J. and Wang, W. [2020]. Neural networks for option pricing and hedging: A literature review, *Journal of Computational Finance* .

Sharpe, W. F. [1966]. Mutual fund performance, *The Journal of Business* **39**(1): 119–138.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A. et al. [2017]. Mastering the game of Go without human knowledge, *Nature* **550**(7676): 354–359.

A Model Evaluation for Puts

This section lists the main evaluation results for puts in a similar way as the results for calls in Section 4.2. As before, we only consider the best hyperparameters of each model, and each training process is repeated 12 times using different random seeds. As in the case of calls, the best hyperparameters for each model are 2 hidden layers and 10 units per hidden layer. For comparison, the corresponding call option tables from Section 4.2 are also included.

Table A1: Average model metrics over train, validation and test sets.

(a) Puts

Model	RMSE	MAPE	errSpread	pSpread
Black-Scholes	train : 1.40 valid : 2.81 test : 2.92	train : 0.98 valid : 1.18 test : 1.06	train : 0.50 valid : 0.80 test : 0.88	train : 68.95 valid : 58.36 test : 55.37
Corrado-Su	train : 1.39 valid : 2.74 test : 2.96	train : 0.87 valid : 1.05 test : 0.97	train : 0.46 valid : 0.73 test : 0.82	train : 71.27 valid : 60.88 test : 57.10
Heston	train : 1.18 valid : 3.34 test : 3.15	train : 0.70 valid : 1.07 test : 0.94	train : 0.38 valid : 0.74 test : 0.79	train : 77.11 valid : 65.26 test : 61.34
Bates	train : 1.14 valid : 2.92 test : 2.90	train : 0.65 valid : 0.96 test : 0.86	train : 0.35 valid : 0.67 test : 0.73	train : 78.77 valid : 67.58 test : 63.68
Avg.	train : 1.28 valid : 2.95 test : 2.98	train : 0.80 valid : 1.07 test : 0.96	train : 0.43 valid : 0.74 test : 0.80	train : 74.02 valid : 63.02 test : 59.37

(b) Calls

Model	RMSE	MAPE	errSpread	pSpread
Black-Scholes	train : 0.80 valid : 2.35 test : 2.79	train : 2.61 valid : 2.81 test : 1.74	train : 0.60 valid : 0.93 test : 1.01	train : 65.51 valid : 53.95 test : 50.24
Corrado-Su	train : 0.70 valid : 2.11 test : 2.07	train : 2.37 valid : 2.50 test : 1.45	train : 0.52 valid : 0.81 test : 0.83	train : 70.29 valid : 57.99 test : 55.11
Heston	train : 0.66 valid : 2.86 test : 3.17	train : 1.74 valid : 2.10 test : 1.42	train : 0.43 valid : 0.81 test : 0.88	train : 74.54 valid : 62.05 test : 58.51
Bates	train : 0.63 valid : 2.90 test : 3.08	train : 1.58 valid : 1.96 test : 1.34	train : 0.40 valid : 0.78 test : 0.85	train : 76.50 valid : 63.86 test : 59.70
Avg.	train : 0.70 valid : 2.55 test : 2.78	train : 2.08 valid : 2.34 test : 1.49	train : 0.49 valid : 0.83 test : 0.89	train : 71.71 valid : 59.46 test : 55.89

Table A2: Average test metrics for different years.

(a) Puts

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
2018	RMSE : 1.37 MAPE : 1.13 errSpread : 0.55 pSpread : 69.48	RMSE : 1.21 MAPE : 0.92 errSpread : 0.43 pSpread : 72.19	RMSE : 1.06 MAPE : 0.74 errSpread : 0.35 pSpread : 79.89	RMSE : 1.03 MAPE : 0.70 errSpread : 0.33 pSpread : 81.35	RMSE : 1.17 MAPE : 0.87 errSpread : 0.42 pSpread : 75.73
2019	RMSE : 1.18 MAPE : 1.00 errSpread : 0.75 pSpread : 55.36	RMSE : 0.97 MAPE : 0.80 errSpread : 0.60 pSpread : 60.20	RMSE : 0.74 MAPE : 0.59 errSpread : 0.44 pSpread : 69.76	RMSE : 0.71 MAPE : 0.55 errSpread : 0.43 pSpread : 70.87	RMSE : 0.90 MAPE : 0.73 errSpread : 0.56 pSpread : 64.05
2020	RMSE : 9.91 MAPE : 2.23 errSpread : 1.92 pSpread : 33.13	RMSE : 10.36 MAPE : 2.21 errSpread : 1.88 pSpread : 34.04	RMSE : 11.85 MAPE : 2.47 errSpread : 1.99 pSpread : 36.29	RMSE : 10.88 MAPE : 2.27 errSpread : 1.89 pSpread : 38.31	RMSE : 10.75 MAPE : 2.29 errSpread : 1.92 pSpread : 35.44
2021	RMSE : 1.26 MAPE : 0.58 errSpread : 0.73 pSpread : 50.21	RMSE : 1.30 MAPE : 0.53 errSpread : 0.68 pSpread : 51.87	RMSE : 1.30 MAPE : 0.58 errSpread : 0.75 pSpread : 49.52	RMSE : 1.15 MAPE : 0.50 errSpread : 0.63 pSpread : 54.78	RMSE : 1.25 MAPE : 0.55 errSpread : 0.70 pSpread : 51.60
2022	RMSE : 0.86 MAPE : 0.36 errSpread : 0.45 pSpread : 68.67	RMSE : 0.97 MAPE : 0.38 errSpread : 0.48 pSpread : 67.20	RMSE : 0.80 MAPE : 0.32 errSpread : 0.42 pSpread : 71.25	RMSE : 0.73 MAPE : 0.30 errSpread : 0.39 pSpread : 73.08	RMSE : 0.84 MAPE : 0.34 errSpread : 0.43 pSpread : 70.05
Avg.	RMSE : 2.92 MAPE : 1.06 errSpread : 0.88 pSpread : 55.37	RMSE : 2.96 MAPE : 0.97 errSpread : 0.82 pSpread : 57.10	RMSE : 3.15 MAPE : 0.94 errSpread : 0.79 pSpread : 61.34	RMSE : 2.90 MAPE : 0.86 errSpread : 0.73 pSpread : 63.68	RMSE : 2.98 MAPE : 0.96 errSpread : 0.80 pSpread : 59.37

(b) Calls

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
2018	RMSE : 0.80 MAPE : 1.77 errSpread : 0.52 pSpread : 66.12	RMSE : 0.69 MAPE : 1.57 errSpread : 0.45 pSpread : 70.67	RMSE : 0.65 MAPE : 1.28 errSpread : 0.38 pSpread : 76.11	RMSE : 0.63 MAPE : 1.19 errSpread : 0.36 pSpread : 77.72	RMSE : 0.69 MAPE : 1.45 errSpread : 0.43 pSpread : 72.66
2019	RMSE : 0.75 MAPE : 1.90 errSpread : 0.81 pSpread : 54.14	RMSE : 0.71 MAPE : 1.72 errSpread : 0.74 pSpread : 56.60	RMSE : 0.60 MAPE : 1.36 errSpread : 0.60 pSpread : 62.28	RMSE : 0.55 MAPE : 1.18 errSpread : 0.54 pSpread : 65.62	RMSE : 0.65 MAPE : 1.54 errSpread : 0.67 pSpread : 59.66
2020	RMSE : 9.87 MAPE : 3.03 errSpread : 2.22 pSpread : 24.50	RMSE : 6.89 MAPE : 2.29 errSpread : 1.72 pSpread : 28.51	RMSE : 12.69 MAPE : 3.09 errSpread : 2.32 pSpread : 28.25	RMSE : 12.34 MAPE : 2.99 errSpread : 2.27 pSpread : 28.52	RMSE : 10.45 MAPE : 2.85 errSpread : 2.13 pSpread : 27.44
2021	RMSE : 1.42 MAPE : 1.49 errSpread : 0.99 pSpread : 42.19	RMSE : 1.22 MAPE : 1.26 errSpread : 0.82 pSpread : 47.28	RMSE : 1.16 MAPE : 1.03 errSpread : 0.71 pSpread : 50.30	RMSE : 1.12 MAPE : 1.01 errSpread : 0.70 pSpread : 50.55	RMSE : 1.23 MAPE : 1.20 errSpread : 0.80 pSpread : 47.58
2022	RMSE : 1.12 MAPE : 0.51 errSpread : 0.53 pSpread : 64.26	RMSE : 0.85 MAPE : 0.41 errSpread : 0.41 pSpread : 72.49	RMSE : 0.74 MAPE : 0.35 errSpread : 0.37 pSpread : 75.62	RMSE : 0.76 MAPE : 0.35 errSpread : 0.37 pSpread : 76.09	RMSE : 0.87 MAPE : 0.40 errSpread : 0.42 pSpread : 72.11
Avg.	RMSE : 2.79 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 2.07 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.17 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 3.08 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.78 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89

Table A3: Average test metrics for different maturity levels.

(a) Puts

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
t1	RMSE : 2.35 MAPE : 1.95 errSpread : 1.30 pSpread : 36.99	RMSE : 2.37 MAPE : 1.63 errSpread : 1.09 pSpread : 41.32	RMSE : 2.44 MAPE : 1.42 errSpread : 0.97 pSpread : 49.38	RMSE : 2.39 MAPE : 1.33 errSpread : 0.90 pSpread : 50.32	RMSE : 2.39 MAPE : 1.58 errSpread : 1.07 pSpread : 44.50
t2	RMSE : 2.28 MAPE : 0.88 errSpread : 0.85 pSpread : 52.15	RMSE : 2.58 MAPE : 0.87 errSpread : 0.86 pSpread : 54.17	RMSE : 2.65 MAPE : 0.85 errSpread : 0.85 pSpread : 56.14	RMSE : 2.53 MAPE : 0.81 errSpread : 0.81 pSpread : 59.69	RMSE : 2.51 MAPE : 0.85 errSpread : 0.84 pSpread : 55.54
t3	RMSE : 2.49 MAPE : 0.80 errSpread : 0.80 pSpread : 59.10	RMSE : 2.71 MAPE : 0.80 errSpread : 0.81 pSpread : 58.87	RMSE : 2.76 MAPE : 0.76 errSpread : 0.78 pSpread : 63.05	RMSE : 2.62 MAPE : 0.72 errSpread : 0.73 pSpread : 66.01	RMSE : 2.65 MAPE : 0.77 errSpread : 0.78 pSpread : 61.76
t4	RMSE : 2.77 MAPE : 0.85 errSpread : 0.89 pSpread : 59.51	RMSE : 2.87 MAPE : 0.81 errSpread : 0.85 pSpread : 59.58	RMSE : 2.98 MAPE : 0.77 errSpread : 0.79 pSpread : 64.76	RMSE : 2.78 MAPE : 0.72 errSpread : 0.75 pSpread : 67.16	RMSE : 2.85 MAPE : 0.79 errSpread : 0.82 pSpread : 62.75
t5	RMSE : 3.20 MAPE : 0.92 errSpread : 0.87 pSpread : 60.03	RMSE : 3.04 MAPE : 0.82 errSpread : 0.78 pSpread : 61.39	RMSE : 3.46 MAPE : 0.89 errSpread : 0.82 pSpread : 65.25	RMSE : 3.04 MAPE : 0.78 errSpread : 0.74 pSpread : 66.66	RMSE : 3.19 MAPE : 0.85 errSpread : 0.80 pSpread : 63.33
t6	RMSE : 3.56 MAPE : 0.95 errSpread : 0.57 pSpread : 64.45	RMSE : 3.47 MAPE : 0.88 errSpread : 0.51 pSpread : 67.28	RMSE : 3.89 MAPE : 0.96 errSpread : 0.51 pSpread : 69.48	RMSE : 3.33 MAPE : 0.83 errSpread : 0.46 pSpread : 72.22	RMSE : 3.56 MAPE : 0.90 errSpread : 0.51 pSpread : 68.36
Avg.	RMSE : 2.77 MAPE : 1.06 errSpread : 0.88 pSpread : 55.37	RMSE : 2.84 MAPE : 0.97 errSpread : 0.82 pSpread : 57.10	RMSE : 3.03 MAPE : 0.94 errSpread : 0.79 pSpread : 61.34	RMSE : 2.78 MAPE : 0.86 errSpread : 0.73 pSpread : 63.68	RMSE : 2.86 MAPE : 0.96 errSpread : 0.80 pSpread : 59.37

(b) Calls

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
t1	RMSE : 1.88 MAPE : 3.56 errSpread : 1.52 pSpread : 30.50	RMSE : 1.85 MAPE : 3.14 errSpread : 1.29 pSpread : 35.86	RMSE : 2.53 MAPE : 2.71 errSpread : 1.23 pSpread : 39.08	RMSE : 2.39 MAPE : 2.49 errSpread : 1.15 pSpread : 40.95	RMSE : 2.16 MAPE : 2.98 errSpread : 1.30 pSpread : 36.60
t2	RMSE : 1.71 MAPE : 1.55 errSpread : 1.03 pSpread : 45.86	RMSE : 1.44 MAPE : 1.34 errSpread : 0.84 pSpread : 51.43	RMSE : 2.53 MAPE : 1.45 errSpread : 0.99 pSpread : 51.64	RMSE : 2.50 MAPE : 1.38 errSpread : 1.00 pSpread : 53.29	RMSE : 2.04 MAPE : 1.43 errSpread : 0.97 pSpread : 50.55
t3	RMSE : 1.94 MAPE : 1.37 errSpread : 0.94 pSpread : 53.18	RMSE : 1.46 MAPE : 1.15 errSpread : 0.76 pSpread : 57.18	RMSE : 2.55 MAPE : 1.14 errSpread : 0.84 pSpread : 61.23	RMSE : 2.48 MAPE : 1.06 errSpread : 0.81 pSpread : 61.93	RMSE : 2.11 MAPE : 1.18 errSpread : 0.84 pSpread : 58.38
t4	RMSE : 2.28 MAPE : 1.28 errSpread : 0.99 pSpread : 53.13	RMSE : 1.64 MAPE : 1.02 errSpread : 0.79 pSpread : 57.57	RMSE : 2.77 MAPE : 1.05 errSpread : 0.87 pSpread : 62.44	RMSE : 2.66 MAPE : 0.97 errSpread : 0.81 pSpread : 63.30	RMSE : 2.34 MAPE : 1.08 errSpread : 0.87 pSpread : 59.11
t5	RMSE : 3.02 MAPE : 1.24 errSpread : 0.96 pSpread : 58.10	RMSE : 2.14 MAPE : 0.99 errSpread : 0.76 pSpread : 61.95	RMSE : 3.32 MAPE : 1.04 errSpread : 0.83 pSpread : 66.23	RMSE : 3.19 MAPE : 1.00 errSpread : 0.81 pSpread : 67.81	RMSE : 2.92 MAPE : 1.07 errSpread : 0.84 pSpread : 63.52
t6	RMSE : 4.29 MAPE : 1.44 errSpread : 0.64 pSpread : 60.66	RMSE : 2.96 MAPE : 1.06 errSpread : 0.51 pSpread : 66.65	RMSE : 4.31 MAPE : 1.15 errSpread : 0.52 pSpread : 70.45	RMSE : 4.24 MAPE : 1.13 errSpread : 0.52 pSpread : 70.92	RMSE : 3.95 MAPE : 1.19 errSpread : 0.55 pSpread : 67.17
Avg.	RMSE : 2.52 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 1.92 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.00 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 2.91 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.59 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89

Table A4: Average test metrics for different strike/moneyness levels.

(a) Puts

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
itm3	RMSE : 3.37 MAPE : 0.77 errSpread : 0.67 pSpread : 64.70	RMSE : 3.41 MAPE : 0.75 errSpread : 0.67 pSpread : 64.51	RMSE : 3.12 MAPE : 0.64 errSpread : 0.58 pSpread : 70.49	RMSE : 2.96 MAPE : 0.59 errSpread : 0.52 pSpread : 73.85	RMSE : 3.22 MAPE : 0.69 errSpread : 0.61 pSpread : 68.39
itm2	RMSE : 2.64 MAPE : 0.69 errSpread : 0.62 pSpread : 64.60	RMSE : 2.73 MAPE : 0.65 errSpread : 0.59 pSpread : 67.21	RMSE : 2.74 MAPE : 0.56 errSpread : 0.54 pSpread : 72.04	RMSE : 2.57 MAPE : 0.53 errSpread : 0.50 pSpread : 74.19	RMSE : 2.67 MAPE : 0.60 errSpread : 0.56 pSpread : 69.51
itm1	RMSE : 2.51 MAPE : 0.73 errSpread : 0.76 pSpread : 58.94	RMSE : 2.69 MAPE : 0.71 errSpread : 0.71 pSpread : 60.78	RMSE : 2.72 MAPE : 0.61 errSpread : 0.63 pSpread : 66.31	RMSE : 2.56 MAPE : 0.59 errSpread : 0.61 pSpread : 68.26	RMSE : 2.62 MAPE : 0.66 errSpread : 0.68 pSpread : 63.57
atm	RMSE : 2.59 MAPE : 0.86 errSpread : 0.85 pSpread : 53.95	RMSE : 2.73 MAPE : 0.84 errSpread : 0.83 pSpread : 53.99	RMSE : 2.91 MAPE : 0.78 errSpread : 0.75 pSpread : 61.09	RMSE : 2.70 MAPE : 0.75 errSpread : 0.73 pSpread : 61.59	RMSE : 2.73 MAPE : 0.81 errSpread : 0.79 pSpread : 57.65
otm1	RMSE : 2.66 MAPE : 1.12 errSpread : 0.99 pSpread : 50.51	RMSE : 2.68 MAPE : 1.01 errSpread : 0.90 pSpread : 53.99	RMSE : 3.05 MAPE : 1.04 errSpread : 0.91 pSpread : 55.94	RMSE : 2.72 MAPE : 0.93 errSpread : 0.83 pSpread : 58.01	RMSE : 2.78 MAPE : 1.03 errSpread : 0.91 pSpread : 54.61
otm2	RMSE : 2.75 MAPE : 1.40 errSpread : 1.05 pSpread : 50.36	RMSE : 2.67 MAPE : 1.21 errSpread : 0.93 pSpread : 52.26	RMSE : 3.15 MAPE : 1.30 errSpread : 0.99 pSpread : 54.51	RMSE : 2.79 MAPE : 1.17 errSpread : 0.91 pSpread : 57.10	RMSE : 2.84 MAPE : 1.27 errSpread : 0.97 pSpread : 53.56
otm3	RMSE : 2.89 MAPE : 1.84 errSpread : 1.22 pSpread : 44.53	RMSE : 2.82 MAPE : 1.61 errSpread : 1.08 pSpread : 46.97	RMSE : 3.42 MAPE : 1.65 errSpread : 1.12 pSpread : 49.02	RMSE : 3.06 MAPE : 1.50 errSpread : 1.03 pSpread : 52.76	RMSE : 3.05 MAPE : 1.65 errSpread : 1.11 pSpread : 48.32
Avg.	RMSE : 2.77 MAPE : 1.06 errSpread : 0.88 pSpread : 55.37	RMSE : 2.82 MAPE : 0.97 errSpread : 0.82 pSpread : 57.10	RMSE : 3.01 MAPE : 0.94 errSpread : 0.79 pSpread : 61.34	RMSE : 2.77 MAPE : 0.86 errSpread : 0.73 pSpread : 63.68	RMSE : 2.84 MAPE : 0.96 errSpread : 0.80 pSpread : 59.37

(b) Calls

	Black-Scholes	Corrado-Su	Heston	Bates	Avg.
itm3	RMSE : 3.06 MAPE : 0.58 errSpread : 0.76 pSpread : 65.49	RMSE : 2.52 MAPE : 0.47 errSpread : 0.62 pSpread : 68.61	RMSE : 4.13 MAPE : 0.68 errSpread : 0.81 pSpread : 69.30	RMSE : 4.11 MAPE : 0.66 errSpread : 0.78 pSpread : 70.82	RMSE : 3.46 MAPE : 0.60 errSpread : 0.74 pSpread : 68.56
itm2	RMSE : 2.74 MAPE : 0.61 errSpread : 0.77 pSpread : 61.32	RMSE : 2.23 MAPE : 0.51 errSpread : 0.67 pSpread : 64.96	RMSE : 3.68 MAPE : 0.71 errSpread : 0.81 pSpread : 65.93	RMSE : 3.58 MAPE : 0.68 errSpread : 0.78 pSpread : 67.00	RMSE : 3.06 MAPE : 0.63 errSpread : 0.76 pSpread : 64.80
itm1	RMSE : 2.79 MAPE : 0.76 errSpread : 0.87 pSpread : 56.57	RMSE : 2.07 MAPE : 0.59 errSpread : 0.71 pSpread : 61.09	RMSE : 3.30 MAPE : 0.78 errSpread : 0.84 pSpread : 63.17	RMSE : 3.19 MAPE : 0.75 errSpread : 0.80 pSpread : 65.12	RMSE : 2.84 MAPE : 0.72 errSpread : 0.81 pSpread : 61.49
atm	RMSE : 2.76 MAPE : 1.00 errSpread : 0.97 pSpread : 51.35	RMSE : 1.91 MAPE : 0.74 errSpread : 0.74 pSpread : 58.29	RMSE : 2.93 MAPE : 0.90 errSpread : 0.84 pSpread : 60.14	RMSE : 2.89 MAPE : 0.88 errSpread : 0.84 pSpread : 60.58	RMSE : 2.63 MAPE : 0.88 errSpread : 0.85 pSpread : 57.59
otm1	RMSE : 2.62 MAPE : 1.46 errSpread : 1.06 pSpread : 44.16	RMSE : 1.85 MAPE : 1.17 errSpread : 0.85 pSpread : 49.48	RMSE : 2.61 MAPE : 1.13 errSpread : 0.83 pSpread : 57.14	RMSE : 2.57 MAPE : 1.10 errSpread : 0.83 pSpread : 58.14	RMSE : 2.41 MAPE : 1.22 errSpread : 0.89 pSpread : 52.23
otm2	RMSE : 2.29 MAPE : 2.51 errSpread : 1.15 pSpread : 40.88	RMSE : 1.62 MAPE : 2.05 errSpread : 0.92 pSpread : 47.62	RMSE : 2.36 MAPE : 1.91 errSpread : 0.90 pSpread : 52.10	RMSE : 2.23 MAPE : 1.86 errSpread : 0.90 pSpread : 51.29	RMSE : 2.12 MAPE : 2.08 errSpread : 0.97 pSpread : 47.97
otm3	RMSE : 2.25 MAPE : 5.27 errSpread : 1.50 pSpread : 31.89	RMSE : 1.68 MAPE : 4.61 errSpread : 1.29 pSpread : 35.69	RMSE : 2.23 MAPE : 3.85 errSpread : 1.11 pSpread : 41.81	RMSE : 2.05 MAPE : 3.47 errSpread : 1.03 pSpread : 44.95	RMSE : 2.05 MAPE : 4.30 errSpread : 1.23 pSpread : 38.59
Avg.	RMSE : 2.64 MAPE : 1.74 errSpread : 1.01 pSpread : 50.24	RMSE : 1.98 MAPE : 1.45 errSpread : 0.83 pSpread : 55.11	RMSE : 3.03 MAPE : 1.42 errSpread : 0.88 pSpread : 58.51	RMSE : 2.95 MAPE : 1.34 errSpread : 0.85 pSpread : 59.70	RMSE : 2.65 MAPE : 1.49 errSpread : 0.89 pSpread : 55.89